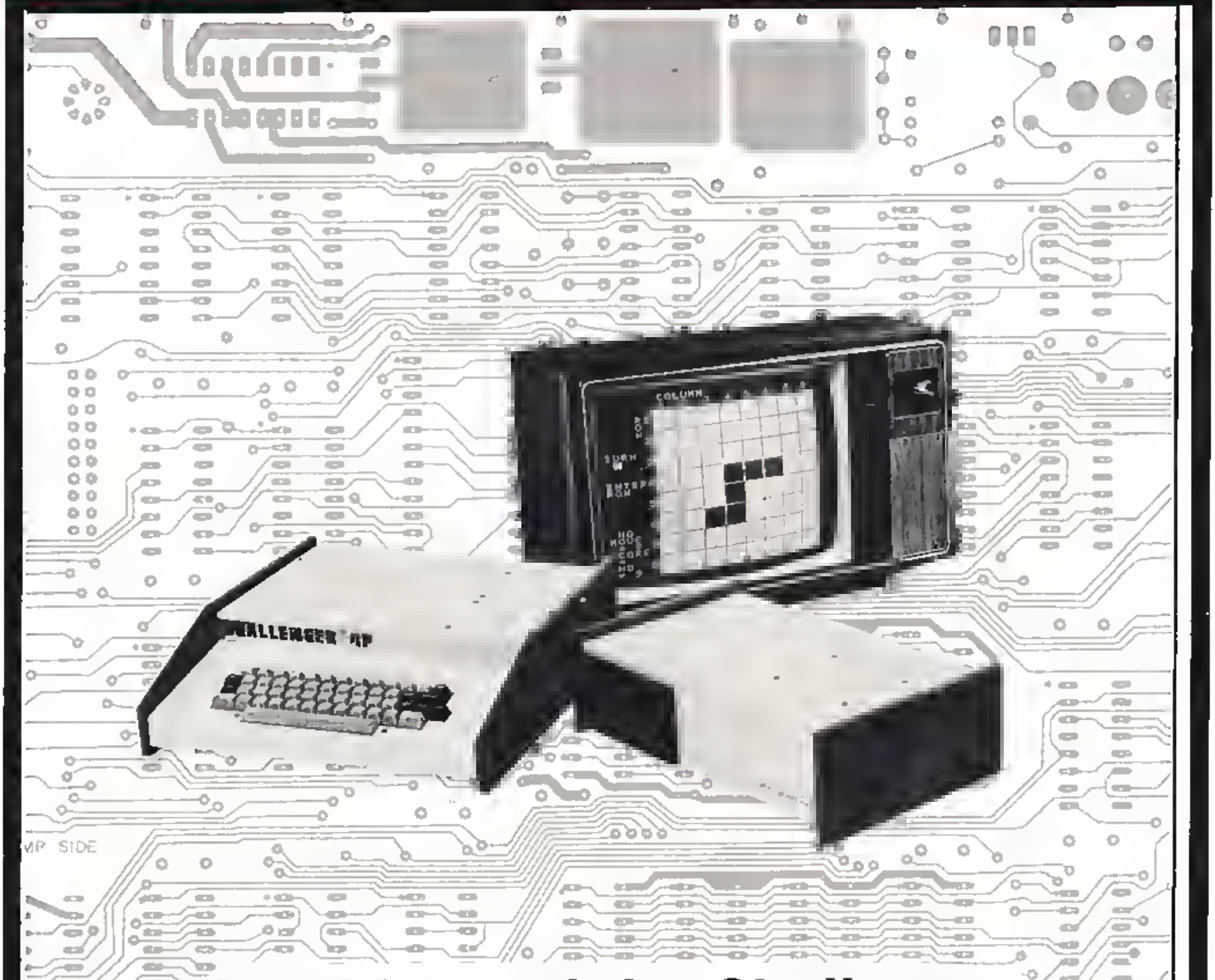
MICRO

THE 6502 JOURNAL



Graphics and the Challenger

NO. 21 FEBRUARY 1980 $2.00

MICRO™

PO Box 6502
Chelmsford, Mass 01824

617-256-5515

"The BEST of MICRO Volume 1" contains all of the important material from the first six issues of **MICRO** in book form.

"The BEST of MICRO Volume 2" contains all of the important material from the second six issues [#7 to 12] of **MICRO** in book form.

"ALL of MICRO Volume 2" is all six issues of Volume 2, issues 7 to 12, at a special reduced price for a limited time while supplies last.

Subscription: One Year = 12 issues. Circle correct category and write amount in space provided.

Surface:		
United States	$15.00	
All Other Countries	$18.00	
Air Mail:		
Central America	$27.00	
Europe/So. America	$33.00	
All Other Countries	$39.00	$ ....
"BEST of MICRO Volume 1"		
Surface	$7.00	
Air Mail	$10.00	$ ....
"BEST of MICRO Volume 2"		
Surface	$9.00	
Air Mail	$13.00	$ ....
"ALL of MICRO Volume 2"		
Surface	$9.00	
Air Mail	$13.00	$ ....

Back Issues:

Issues 7 to 12:	No.	Surface @ $1.75 each Air Mail @$2.75 each	=	$ ....
Issues 13 on:	No.	Surface @ $2.25 each Air Mail @ $3.25 each	=	$ ....

All payments must be in US dollars.
Make checks payable to: MICRO
Foreign payments in International Money Order or cash.

TOTAL $ ....

If you are a subscriber, attach label or write subscription number here: ....

Name: ....

Address: ....

....

City: .... State: .... Zip: ....

Country (if not U.S.): ....

Help **MICRO** bring you the info you want by completing this short questionnaire.

Microcomputers Owned/Planning to Buy: AIM SYM KIM PET APPLE OSI Other: ....

Peripherals Owned/Planning to Buy: Memory Disk Video Printer Terminal Other: ....

Microcomputer Usage: Educational Business Personal Control Games Other: ....

Languages Used: Assembler BASIC FORTH PASCAL Other: ....

Your comments and suggestions on **MICRO**: ....

....

Club/Group User Registration Form

Name: .... President: ....

Location: ....

No. of Members: ....

Meeting algorithm (date, time, place): ....

....

....

Publications: ....

....

Aim/Purpose of the group: ....

....

For Current Information, Contact: ....

....

....

Software/Hardware Catalogue Entry

Do you have a software or hardware package you want publicized? Our Software and Hardware Catalogues offer a good opportunity to receive some free advertisement. These regular features of MICRO are provided both as a service to our readers and as a service to the 6502 industry which is working hard to develop new and better software and hardware products for the 6502 based system. There is no charge for listings in these catalogues. All that is required is that material for the listing be submitted in the listing format. All information should be included. We reserve the right to edit and/or reject any submission. We might not edit the description the same way you would, so please, be brief and specific.

Name:..

System:..

Memory:..

Language:..

Hardware:..

Description:..

....

....

....

....

....

....

Copies:..

Price:..

Author:..

Available from:..

....

....

Classified Ad

Classified ads provide an economical way to announce new products or sales promotions, generate product interest, enhance visibility and promote good will. MICRO clusters large format classified ads at high impact locations throughout each issue. Because classifieds represent a service to readers, MICRO must restrict each advertiser to a single, six-line insertion per issue. The nominal $10 charge reflects our preparation costs and must be prepaid.

Description:..

....

....

....

....

....

....

Name:..

Address:..

City,State:........................Zip: ....

Other

If you are interested in Writing for MICRO, becoming a MICRO dealer or advertising in MICRO, please indicate below and the information will be mailed to you.

Dealer Information Package: □

Advertiser's Media Package: □

MICRO Writer's Guide: □

....

First
Class
Stamp

MICRO™

P.O. Box 6502
Chelmsford, MA 01824

"As manager of Personal Computers, here at Computerland of San Francisco, evaluating new software products is part of my job. With all the word processors on the market today, I choose EasyWriter for my business and personal use."
—Karen Dexter Weiss

EasyWriter T.M.

80 COLUMNS OF WORD PROCESSING POWER

FOR YOUR APPLE II COMPUTER

Finally . . . INFORMATION UNLIMITED SOFTWARE is able to bring you a complete word processing system for the Apple II. The new EasyWriter system gives you ***80 columns of upper and lower case characters*** for your Apple's video display, using the new SUP'R'TERM 1 board!

A long time ago, we decided to bring you the best simple-to-use and understand tools for your computer system. Today we've taken another stride in that same direction. It took some doing, in both hardware and software, but we think you'll agree that for the buck, no one can touch us.

Check it out:

- 80 Columns on the Screen!
- Upper & Lower Case!
- Global Search & Replace!
- Underlining!
- Bidirectional Printing!
- Incremental Spacing!
- File Appending!
- 50 Pages of Text Per Disk!

You can purchase the new EasyWriter 80 column word processing system as a complete hardware and software package directly from our new office in California, or from your local computer dealer.

Information
Unlimited
Software™ Inc.

Information Unlimited Software, Inc.
793 Vincente St.
Berkeley, CA 94707
(415) 525-4046

- EasyWriter is a TM of Cap'n Software, Inc.
- Apple II is a TM of Apple Computers, Inc.

See The System at the 5th West Coast Computer Faire.

softside software
$15.95
PET
ASSEMBLER 2001
full featured mnemonic assembler-disassembler
enter save load list run
list the secret basic ROMs

PET BASIC BREAKTHROUGH

Softside Software presents

SYMBOLIC/STRUCTURED BASIC

At last, **Symbolic/Structured Basic** is available for your PET 8-32K personal computer! S-Basic is a pre-compiler that enhances the PET's built-in basic monitor with the addition of extra-control statements found only in the most sophisticated computers. **WHILE** ... **GOSUB** ... calls a subroutine as long as a condition is true. **UNTIL ... GOSUB ...** jumps to a subroutine unless the condition is true. The **IF** ... **Then** ... **ELSE** statement allows the programmer to command the computer to execute instructions if the normal IF condition is not met.

Forget about line numbers, S-Basic allows you to program naturally only naming (numerically or alphabetically!) statements that you will need to refer to, for example: LOOP/PRINT "HI": GO TO LOOP. S-Basic program **lines can be up to 255 characters long**, two-and-one-half times as long as on standard Basic. S-Basic does not compromise any of PET Basic's existing features. All PET Basic commands can be used.

S-Basic includes an **editor** with full text capabilities, a **translator/pre-compiler** with its own error messages, and the **S-Basic loader**. These programs are recommended for disk-based PETs. A printer is optional but suggested. Cassette copies are available and require two cassette drives. Comprehensive instructions are included. Symbolic-/Structured Basic package is available complete for an **introductory price** of **$35.95**.

- WHILE....GOTO....
- IF....THEN....ELSE
- 255 CHAR. LINES

A PET PROGRAMMING BREAKTHROUGH

- UNTIL....GOTO....
- SYMBOLIC OPTIONAL
- LINE NUMBERS

305 RIVERSIDE DRIVE, NEW YORK, NEW YORK 10025

MICRO™

February 1980
Issue Number 21

Table of Contents

Staff

Editor/Publisher
Robert M. Tripp

Associate Editor
Mary Ann Curtis

Assistant Editor
Evelyn M. Heinrich

Business Manager
Maggie E. Fisher

Circulation Manager
Carol A. Stark

Comptroller
Donna M. Tripp

Production Assistant
L. Catherine Bland

MICRO Lab
R. Keith Beal

MICRO™ is published monthly by MICRO INK, Inc., Chelmsford, MA 01824. Tel. 617/256-5515.

Second Class postage paid at Chelmsford, Ma 01824.

Publication Number: COTR 395770.

Circulation: Paid subscriptions: U.S.: 3700, Foreign: 350; Dealers: U.S.: 3500, Foreign: 1700.

Subscription rates: U.S.: $15 per year. Foreign, surface mail: $18 per year.

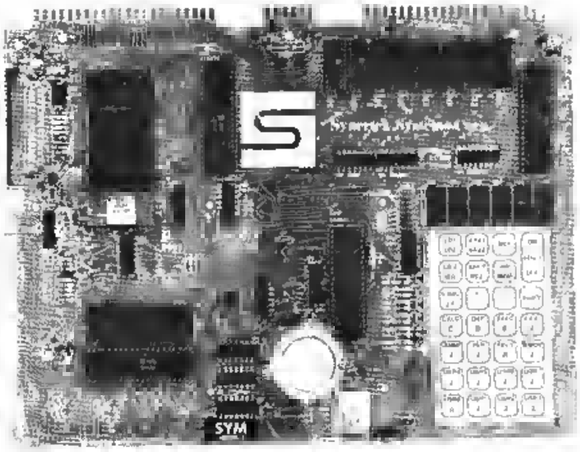
For air mail rate, change of address, back issue or subscription information write to: MICRO, P.O. Box 6502, Chelmsford, MA 01824.

Advertiser's Index

SYM-1, 6502-BASED MICROCOMPUTER

- FULLY-ASSEMBLED AND COMPLETELY INTEGRATED SYSTEM that's ready-to-use
- ALL LSI IC'S ARE IN SOCKETS
- 28 DOUBLE-FUNCTION KEYPAD INCLUDING UP TO 24 "SPECIAL" FUNCTIONS
- EASY-TO-VIEW 6-DIGIT HEX LED DISPLAY
- KIM-1* HARDWARE COMPATIBILITY

  The powerful 6502 8-Bit MICROPROCESSOR whose advanced architectural features have made it one of the largest selling "micros" on the market today.
- THREE ON-BOARD PROGRAMMABLE INTERVAL TIMERS available to the user, expandable to five on-board.
- 4K BYTE ROM RESIDENT MONITOR and Operating Programs.
- Single 5 Volt power supply is all that is required.
- 1K BYTES OF 2114 STATIC RAM onboard with sockets provided for immediate expansion to 4K bytes onboard, with total memory expansion to 65, 536 bytes.
- USER PROM/ROM: The system is equipped with 3 PROM/ROM expansion sockets for 2316/2332 ROMs or 2716 EPROMs
- ENHANCED SOFTWARE with simplified user interface
- STANDARD INTERFACES INCLUDE:
  - —Audio Cassette Recorder Interface with Remote Control (Two modes: 135 Baud KIM-1* compatible, Hi-Speed 1500 Baud)
  - —Full duplex 20mA Teletype Interface
  - —System Expansion Bus Interface
  - —TV Controller Board Interface
  - —CRT Compatible Interface (RS-232)
- APPLICATION PORT: 15 Bi-directional TTL Lines for user applications with expansion capability for added lines
- EXPANSION PORT FOR ADD-ON MODULES (51 I/O Lines included in the basic system)
- SEPARATE POWER SUPPLY connector for easy disconnect of the d-c power
- AUDIBLE RESPONSE KEYPAD



Synertek has enhanced KIM-1* software as well as the hardware. The software has simplified the user interface. The basic SYM-1 system is programmed in machine language. Monitor status is easily accessible, and the monitor gives the keypad user the same full functional capability of the TTY user. The SYM-1 has everything the KIM-1* has to offer, plus so much more that we cannot begin to tell you here. So, if you want to know more, the SYM-1 User Manual is available, separately.

SYM-1 Complete w/manuals	**$269.00**
SYM-1 User Manual Only	**7.00**
SYM-1 Expansion Kit	**75.00**

Expansion includes 3K of 2114 RAM chips and 1-6522 I/O chip.

SYM-1 Manuals: The well organized documentation package is complete and easy-to-understand.

SYM-1 CAN GROW AS YOU GROW. Its the system to BUILD-ON. Expansion features that are soon to be offered:

***BAS-1 8K Basic ROM (Microsoft)**	**$159.00**
***KTM-2 TV Interface Board**	**349.00**

We do honor Synertek discount coupons

QUALITY EXPANSION BOARDS DESIGNED SPECIFICALLY FOR KIM-1, SYM-1 & AIM 65

These boards are set up for use with a regulated power supply such as the one below, but, provisions have been made so that you can add onboard regulators for use with an unregulated power supply. But, because of unreliability, we do not recommend the use of onboard regulators. All I.C.'s are socketed for ease of maintenance. *All boards carry full 90-day warranty.*

All products that we manufacture are designed to meet or exceed industrial standards. All components are first qualtiy and meet full manufacturer's specifications. All this and an extended burn-in is done to reduce the normal percentage of field failures by up to 75%. To you, this means the chance of inconvenience and lost time due to a failure is very rare; but, if it should happen, we guarantee a turn-around time of less than forty-eight hours for repair.

Our money back guarantee: If, for any reason you wish to return any board that you have purchased directly from us within ten (10) days after receipt, complete, in original condition, and in original shipping carton; we will give you a complete credit or refund less a $10.00 restocking charge per board.

VAK-1 8-SLOT MOTHERBOARD

This motherboard uses the KIM-4* bus structure. It provides eight (8) expansion board sockets with rigid card cage. Separate jacks for audio cassette, TTY and power supply are provided. Fully buffered bus.

VAK-1 Motherboard	**$129.00**

VAK-2/4 16K STATIC RAM BOARD

This board using 2114 RAMs is configured in two (2) separately addressable 8K blocks with individual write-protect switches.

VAK-2 16K RAM Board with only 8K of RAM (½ populated)	**$239.00**
VAK-3 Complete set of chips to expand above board to 16K	**$175.00**
VAK-4 Fully populated 16K RAM	**$379.00**

VAK-5 2708 EPROM PROGRAMMER

This board requires a +5 VDC and ±12 VDC, but has a DC to DC multiplyer so there is no need for an additional power supply. All software is resident in on-board ROM, and has a zero-insertion socket.

VAK-5 2708 EPROM Programmer	**$269.00**

VAK-6 EPROM BOARD

This board will hold 8K of 2708 or 2758, or 16K of 2716 or 2516 EPROMs. *EPROMs not included.*

VAK-6 EPROM Board	**$129.00**

VAK-7 COMPLETE FLOPPY-DISK SYSTEM (May '79)

VAK-8 PROTYPING BOARD

This board allows you to create your own interfaces to plug into the motherboard. Etched circuitry is provided for regulators, address and data bus drivers; with a large area for either wire-wrapped or soldered IC circuitry.

VAK-8 Protyping Board	**$49.00**

POWER SUPPLIES

ALL POWER SUPPLIES are totally enclosed with grounded enclosures for safety, AC power cord, and carry a full 2-year warranty.

FULL SYSTEM POWER SUPPLY

This power supply will handle a microcomputer and up to 65K of our VAK-4 RAM. ADDITIONAL FEATURES ARE: Over voltage Protection on 5 volts, fused, AC on/off switch. Equivalent to units selling for $225.00 or more.

Provides +5 VDC @ 10 Amps & ±12 VDC @ 1 Amp

VAK-EPS Power Supply	**$125.00**

KIM-1* Custom P.S. provides 5 VDC @ 1.2 Amps and +12 VDC @ .1 Amps

KCP-1 Power Supply	**$41.50**

SYM-1 Custom P.S. provides 5 VDC @ 1.4 Amps

VCP-1 Power Supply	**$41.50**

*KIM is a product of MOS Technology

RNB ENTERPRISES
INCORPORATED

2967 W. Fairmount Avenue
Phoenix AZ 85017
(602)265-7564

master charge VISA

Expanding the SYM-1... Adding an ASCII Keyboard

Adding an ASCII keyboard to a SYM is fairly simple, if you know what you are doing. There are a few tricks required and some understanding of the SYM Monitor is needed. And, it is all presented here.

Robert A. Peck
P.O. Box 2231
Sunnyvale, CA 94087

The Synertek monitor program has a teature which allows it to communicate directly with a teletype system. This is, when you are in the reset mode, the monitor will scan both the onboard keypad and the teletype input port to look for the first keystroke. After finding the first stroke, either the keypad or the teletype is used as the exclusive input to the monitor program.

Because ot the teletype intertace, it would, at first thought, be an excellent way to expand the basic SYM system. However, when one considers the bulk, cost and availability of a teletype, other alternatives for early stage expansion may come to mind.

Synertek also otters a keyboard/video display unit tor the SYM-t, known as the KTM-2. It is a very versatile unit; but the present list price of $349 could cause some ot us to wait a bit to budget for its eventual purchase. What then to do in the meantime?

To at least begin a system expansion at a low cost, one might consider adding a full ASCII keyboard now and a full video display as a separate step at a later date. ASCII keyboards are available on the surplus scene for as little at $35, so this seems like a good place to start.

An initial thought in adding the ASCII keyboard to the SYM would be to duplicate the tunctions of the teletype. This would pose a couple ot unwelcome complications, specitically the choice of an appropriate baud rate and the addition of a parallel to serial conversion to the ASCII keyboard output.

However, it we attach the keyboard to the teletype Input and log onto the keyboard, the SYM monitor will respond to us in bit serial mode as well. We would then, at least for a period of time, lose our display capabilities. We would have to restore the onboard display vector in order to see the results ot our keystrokes.

Since a certain amount of software had to be written anyway to bypass the above problem, it seemed appropriate to solve some hardware problems with sottware Instead. I added VIA No. 2 (6522) to the system to provide an extra set of input ports, one of which I dedicated to the parallel ASCII keyboard. Port B is used tor the 6522 timer functions so to preserve these for future use.; Port A was chosen for the keyboard.

In the attempt to add the keyboard to the system, a number ot items were kept in mind:

(A) All ot the monitor tunctions had to be normally accessible (different key groups perhaps, but all tunctions still needed).

(B) The use of the keyboard in place ot the keypad should not intertere with the execution ot any programs I had already written or adapted for use tith the SYM if at all possible.

(C) The Interface routines should be written In a fully relocatable style so that they could be incorporated into a monitor PROM routine it desired.

In keeping with these principles, the program shown in Figure 1 was written to perform the monitor intertacing.

When one desires to use the external ASCII keyboard instead of the keypad, the routine labeled INIT would be executed. A direct jump to this routine is used. It modifies both the keyboard input vector and the keyboard status vector, providing tor entry to the other routines. Then it does a warm start jump back to the main segment of the monitor program.

Following the execution ot the INIT routine, the monitor program will always check the external keyboard tor its inputs. Only the reset key on the keypad is still active at this point. To restore full control to the onboard keypad, one needs only to push reset or execute a jump to location 8B4A which is the beginning of the power-on reset routine (simulates pushing the reset switch).

Now that we've used INIT, let's see what functions we have and how to access them. To begin with, there are two routines in Figure 1 referred to by the INIT program:

GKEY, the equivalent of SYM GETKEY, and

KSTAT, the equivalent of SYM KYSTAT.

Both routines affect the same registers (A,F) and have the same overall effect as noted in the SYM manual, page 9-3.

The KSTAT routine reads the input port addressed as A801, then left-shifts the input byte. If there is an input there, the carry bit will be set. Therefore KSTAT, as a subroutine, performs exactly the same function of KYSTAT.

The ASCII keyboard is connected with its 7 output bits on port A bits 2PA6-2PAO. Port 2PA7 is used for a key strobe input (any key down). The keyboard parity bit, if any, is not used in this application. If no key is down, the input port will be read as all zeros. If any key is down, the most significant bit of the input port will be a one due to the presence of the keystrobe bit, allowing a single left shift to set the carry bit.

The GKEY routine performs the same function as GETKEY in that it scans the display while waiting for a key to be pressed. In the process of waiting for a keystroke, the scanning of the display is controlled through the display scanning vector. This allows the user to make use of the oscilloscope output routine with only minor modifications, substituting a JSR to GKEY for the JSR to GETKEY.

All other specifications mentioned in the Synertek manual for the oscilloscope driver routine will then be valid. As a matter of fact, access to an oscilloscope and the use of the driver routine could temporarily satisfy a person's desire for a video display, at least until some suitable alternative could be found.

The ASCII keyboard scanning routine GKEY handles the keybounce problem by going into a small wait loop immediately after sensing that a key is down, then scans the display while it waits for the key to be released. After release, it interprets the original keystroke contents by stripping off the keystrobe bit and returning to the calling program with the ASCII equivalent of the key in the accumulator.

Now that we've seen how the routines provide for the communication with the new keyboard, lets see how we can access all of the SYM monitor functions without resorting to the use of the keypad.

Because of the direct relation of the ASCII equivalents, the following control functions are directly accessible:

Memory: M
Verify: V
Block move: B
Write protect:W
Register: R
Deposit: D
Jump: J
Execute: E
Go: G
Calculate: C
Fill: F

```
20 88 81   GKEY    JSR    SAVER      SAVE REGISTERS
AD 01 A8           LDA    A801       GET PARALLEL ASCII
F0 24              BEQ    DISP       UNLESS NONE,THEN BRANCH
85 F1              STA    00F1       STORE IT A WHILE
A9 10              LDA    #$10       DEBOUNCE CONSTANT
85 EF              STA    00EF       DEBOUNCE
C6 F0      WAIT1   DEC    00F0       SMALL LOOP
D0 FC              BNE    WAIT1
C6 EF              DEC    00EF       LARGE LOOP
D0 F8              BNE    WAIT1
20 03 89   SCANA   JSR    IJSCNV     SCAN DISPLAY(USE SCANVEC)
2C 01 A8           BIT    A801       IS KEY STILL DOWN?
30 F8              BMI    SCANA      WAIT FOR KEY RELEASE
A5 F1              LDA    00F1       KEY UP, PROCESS KEY
29 7F              AND    #$7F       STRIP KEY STROBE BIT
A5 F1              JSR    OUTCHR     SEND INTO DISPUF
A5 F1              LDA    00F1       GET IT AGAIN
29 7F              AND    #$7F       STRIP IT AGAIN
4C 88 81           JMP    RESXAF     FETURN WITH ASCII IN A
A9 10      WAIT2   LDA    #$10       IF NO KEY,
85 EF              STA    00EF       SCAN DISPLAY
20 03 89   SCANB   JSR    IJSCNV     THRU SCANVEC
C6 EF              DEC    00EF       A NUMBER OF TIMES
D0 F9              BNE    SCANB      THEN GO BACK
F0 CA              BEQ    GKEY       AND LOOK AGAIN
AD 01 A8   KSTAT   LDA    A801       READ ASCII INPORT
0A                 ASLA              SHIFT MSB INTC CARRY
60                 RTS               PET, CFLAG=1 IF KEY DN.

20 86 88   INIT    JSR    ACCESS     UNPROTECT SYSRAM
A9 00              LDA    #00        MODIFY
8D 61 A6           STA    A661       KEYBOARD
A9 02              LDA    #02        INPUT
8D 62 A6           STA    A662       VECTOR
A9 40              LDA    #$40       MODIFY
8D 67 A6           STA    A667       KEYPRESS
A9 02              LDA    #02        STATUS
8D 68 A6           STA    A668       VECTOR
4C 03 80           JMP    WARM       WARM ENTRY,MONITCR
```

Figure 1: **ASCII Keyboard Interface initialization and communication routines.**

Likewise, again because of the direct ASCII usage by the monitor, the carriage return (CR), plus sign, minus sign, forward arrow and reverse arrow functions of the ASCII keyboard will perform the same functions as those equivalent keys on the built-in keypad.

Accessing the remainder of the monitor functions will require the use of two keys simultaneously, in the fashion of a shifted character. One of the keys is the CONTROL key often found on an ASCII keyboard. The function of this key (if your keyboard doesn't have one) is to inhibit the output of the two most significant bits of the ASCII output, in this case,to force a zero to both input lines 2PA6 and 2PA5. This can be accomplished with a single switch and one type 7408 IC as suggested in Figure 2.

The following functions are accessed by first holding down the control key, then pressing the indicated ASCII key: (control key referenced by CNTL below)

Store Double Byte: CNTL P
Load Paper Tape: CNTL Q
LD1 (KIM format): CNTLR LD2 (SYM hi spd): CNTL S
USRO: CNTL T
USR1: CNTL U
USR2: CNTL V
USR3: CNTL W
USR4: CNTL X
USR5: CNTL Y
USR6: CNTL Z
USR7: CNTL (
SAVP save paper tape: CNTL
SAV1 (KIM format): CNTL)
SAV2 (SYM hi spd): CNTL

As may be seen above, although certain of the keys may be different, all of the monitor functions are accessible from the external keyboard, fulfilling our objectives in adding it in the first place. Actually I have hedged a bit for a couple of items, but these items I figure are not needed on the external keyboard, but serve their purpose better on the keypad, specifically the DEBUG ON/OFF, the SHIFT, and the ASCII keypad items. DEBUG is a hardware function which can be simulated by software, so in a program we can access the function. SHIFT is a monitor translation routine, appropriate only to the placement and arrangement of the keys on the keypad. Finally, the ASCII key is not necessary externally since everything we output from the external keyboard is formatted in parallel ASCII anyway.

The SYM-1 is a very powerful single-board computer. The addition of a parallel ASCII keyboard inexpensively provides us with a basis for further expansion of the SYM-1's capabilities.

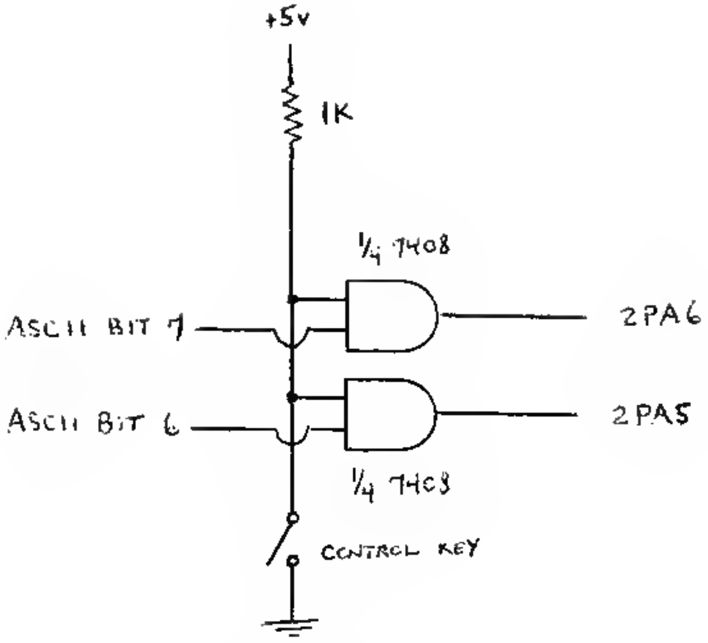


***Figure* 2: Adding a CONTROL key**

The SY6516 PSEUDO-16 microprocessor, after power up, is identical to the 6500 series microprocessors in terms of instruction set (source code only), registers and system timing. However, due to improvements made in the state counter and look ahead carry in the SY6516, several of the instructions in the 6500 series will require fewer cycles to execule. Instructions in this category are:

Instruction	Addressing Mode	6500 #Cycles	6516 #Cycles
STA	(IND,Y)	6	5
	(ABS,X)	5	4
LDA	ABS,Y	4	3
INC	ABS,X	7	6
DEC	ABS,X	7	6
ASL	ABS,X	7	6
ROL	ABS,X	7	6
ROR	ABS,X	7	6
TAX	IMPLIED	2	1
TXA	IMPLIED	2	1
TAY	IMPLIED	2	1
TYA	IMPLIED	2	1
TSX	IMPLIED	2	1
TXS	IMPLIED	2	1
SEC	IMPLIED	2	1
CLC	IMPLIED	2	1
SED	IMPLIED	2	1
CLD	IMPLIED	2	1
SEI	IMPLIED	2	1
CLI	IMPLIED	2	1
CLV	IMPLIED	2	1
INX	IMPLIED	2	1
DEX	IMPLIED	2	1
DEY	IMPLIED	2	1
PLP	IMPLIED	4	3
PLA	IMPLIED	4	3
NOP	IMPLIED	2	1
RTI	IMPLIED	6	5
RTS	IMPLIED	6	4
TSX	FLAGS	N,Z	NO FLAGS
TSR	ABS	6	5

***Table 1:* SY6516 Pseudo-16 compatability to SY6500 series microprocessors**

PROGRESSIVE SOFTWARE

Presents
Software and Hardware for your APPLE

SALES FORECAST provides the best forecast using the four most popular forecasting techniques: linear regression, log trend, power curve trend, and exponential smoothing. Neil D. Lipson's program uses artificial intelligence to determine the best fit and displays all results for manual intervention. **$9.95**

CURVE FIT accepts any number of data points, distributed in any fassion, and fits a curve to the set of points using log curve fit, exponential curve fit, least squares, or a power curve fit. It will compute the best fit or employ a specific type of fit, and display a graph of the result. By Dave Garson. **$9.95**

UTILITY PACK 1 combines four versatile programs by Vince Corselli, for any memory configuration.

- **Integer to Applesoft conversion**: Encounter only those syntax errors unique to Applesoft after using this program to convert any Integer BASIC source.
- **Disk Append**: Merge any two Integer BASIC sources into a single program on disk.
- **Integer BASIC copy**: Replicate an Integer BASIC program from one disk to another, as often as required, with a single keystroke.
- **Applesoft Update**: Modify Applesoft on the disk to eliminate the heading always produced when it is first run.
- **Binary Copy**: Automatically determines the length and starting address of a program while copying its binary file from one disk to another in response to a single keystroke. **$9.95**

MISSILE-ANTI-MISSILE display a target, missile, anti-missile, a submarine and map of the U.S. on the screen. A hostile submarine appears and launches a pre-emptive nuclear attack controlled by paddle 1. As soon as the hostile missile is fired, the U.S. launches its anti-missile controlled by paddle 0. Dave Moteles' program offers high resolution and many levels of play. **$9.95**

TOUCH TYPING TUTOR teaches typing. Indicates speed and errors made. Finger Bldrs, Gen. Typing, Basic Language and User Supplied. Diskette. Written by Wm. A. Massena. **$19.95**

APPLE MENU COOKBOOK index-accessed data storage/retrieval program. Recipes stored, unlimited lines per entry. Easy editing. Formulated after N.Y. Times Cookbook. Other useful features included.
Written by Wm. Merlino, M.D. **$19.95**

MAILING LIST PROGRAM maintains complete record of name, address, phone no., mailing labels accommodates parallel card or built-in printer driver, easy data entry.
Diskette. 32K. **$19.95**

BLOCKADE lets two players compete by building walls to obstruct each other. An exciting game written in Integer BASIC by Vince Corselli. **$9.95**

TABLE GENERATOR forms shape tables with ease from directional vectors and adds additional information such as starting address, length and position of each shape. Murray Summers' Applesoft program will save the shape table anywhere in usable memory. **$9.95**

OTHELLO may be played by one or two players and is similar to chess in strategy. Once a piece has been played, its color may be reversed many times, and there are also sudden reverses of luck. You can win with a single move. Vince Corselli's program does all the work of keeping board details and flipping pieces. **$9.95**

SINGLE DRIVE COPY is a special utility program, written by Vince Corselli in Integer BASIC, that will copy a diskette using only one drive. It is supplied on tape and should be loaded onto a diskette. It automatically adjusts for APPLE memory size and should be used with DOS 3.2. **$19.95**

SAUCER INVASION
SPACE MAZE
STARWARS
ROCKET PILOT Written by Bob Bishop Each **$9.95**

SAUCER INVASION lets you defend the empire by shooting down a flying saucer. You control your position with the paddle while firing your missile at the invader. Written by Bob Bishop. **$9.95**

HARDWARE

LIGHT PEN with seven supporting routines. The light meter takes intensity readings every fraction of a second from 0 to 588. The light graph generates a display of light intensity on the screen. The light pen connects points that have been drawn on the screen, in low or high resolution, and displays their coordinates. A special utility displays any number of points on the screen, for use in menu selection or games, and selects a point when the light pen touches it. The package includes a light pen calculator and light pen TIC TAC TOE. Neil D. Lipson's programs use artificial intelligence and are not confused by outside light. The hi-res light pen, only, requires 48K and ROM card. **$34.95**

POSTAGE AND HANDLING

Please add $1.25 for the first item
and $.75 for each additional item.

- **Programs accepted for publication**
- **Highest royalty paid**

TO ORDER

Send check or money order to:

P.O. Box 273
Plymouth Meeting, PA 19462

PA residents add 6% sales tax.

U.S. and foreign dealer and distributor inquiries invited
All programs require 16K memory unless specified

A HIRES Graph-Plotting Subroutine in Integer BASIC for the APPLE II

A BASIC subroutine is presented which permits HIRES graph plotting. It includes X and Y axes generation with scale markers as well as the plotting of user specified points. This will make it easy to display the results of a variety of problems, functions, correlations, etc.

Richard Fam
36 Eifth Avenue
Singapore 10
Republic of Singapore

The article entitled APPLE II High Resolution Graphics Memory Organization, foung in MICRO 7:43 by Andrew H. Eliason is of tremendous value to those who wish to plot in HIRES graphics. The following graph plotting subroutine utilizes formulae given in this article.

Referring to the listing on being called by the GOSUB 9000 statement in the main program, the subroutine first clears page 1 of HIRES graphics memory at line 9023. This is quite a time-consuming process and the impatient experimenter may care to replace this line with a CALL statement to an equivalent machine language subroutine. I have actually tried this and found that it reduces the time execution for the complete plotting routine by approximately half.

Having set the graphics and HIRES modes in line 9060, the routine then proceeds to plot the X and Y axes. Scale markers are placed at 20-point intervals along the two axes.

The final stage in the subroutine involves the plotting of the points. The magnitude of these points are stored in matrix GPH which is dimensioned for 279 elements in the main program. Only values GPH(X) between 0 and 91 inclusive can be plotted.

As you may recall, the display area of HIRES graphics is a matrix comprised of 280 horizontal by 192 vertical points. The subroutine fetches elements of GPH, does the necessary calculations, and outputs the results on the screen. To prevent the disfigurement of the two axes, I have avoided the plotting of points less than one byte away from the Y-axis and on the X-axis itself.

For successful application of this graph plotting subroutine, observe the following rules:

a) Only an APPLE II with a minimum of 16K bytes of memory can be used

b) Ensure that the main program contains the statement DIM GPH(279).

c) Only values of GPH(X) such that 0 GPH(X) 191 where X ranges from 0 to 279, inclusive, will be plotted.

d) Set HIMEM:8191 to restrain intrusion into page 1 of HIRES graphics memory.

Here are two short programs demonstrating the performance of the high resolution graphics-plotting subroutine.

```
10     DIM GPH(279)
20     FOR I=0 TO 279
30     GPH(I) = RND(191)
40     NEXT I
50     GOSUB 9000
60     END
```

```
10     DIM GPH(279)
20     FOR I=0 TO 279
30     GPH(I)=I/2 — 30
40     NEXT I
50     GOSUB 9000
```

AIR SPD
ALT
TURN RATE HEADING VERT. V
OIL P-63 OIL T-228 FUEL-24 TACH-100

```
LIST
9000 REM  *
9001 REM  * HIRES GRAPH-PLOTTING
9002 REM  * SUBROUTINE
9003 REM  *
9004 REM  * BY R.S.K. FAM
9005 REM  *               26/4/79
9006 REM  *
9007 REM  * DATA IS STORED IN GPH(X)
9008 REM  * CONSISTIJG OF 200 POINTS
9009 REM  * 0 <= GPH(X) <=191
9010 REM  *
9011 REM  * SET HIMEM:8191
9012 REM  *
9020 REM  *
9021 REM  * CLEAR SCREEN
9022 REM  *
9023 FOR I=8192 TO 16383: POKE I,
     0: NEXT I
9030 REM  *
9040 REM  * SET HIRES MODE
9050 REM  *
9060 POKE -16304,0: POKE -16297,
     0: POKE -16302,0
9140 REM  *
9150 REM  * PLOT Y-AXIS
9160 REM  *
9170 FOR LV=0 TO 191:PT=1: IF (LV+
     9) MOD 20=0 THEN PT=7: POKE
     (LV MOD 8*1024+(LV/8) MOD 8
     *128+(LV/64)*40+8192),PT: NEXT
     LV
9200 REM  *
9210 REM  * PLOT X-AXIS
9220 REM  *
9230 PT=0: FOR LH=0 TO 279: IF LH MOD
     20<>0 THEN 9240:PT=PT+1: FOR
     MK=1 TO 2: POKE LH/7+16336-
     (1024*MK),64/(2  ((PT+5) MOD
     7)): NEXT MK: GOTO 9242
9240 POKE LH/7+16336,255
9242 NEXT LH
9260 REM  *
9270 REM  * PLOT POINTS
9280 REM  *
9290 FOR LH=8 TO 279:LV=191-GPH(
     LH): IF LV<0 OR LV>=191 THEN
     9330
9310 BV=LV MOD 8*1024+(LV/8) MOD
     8*128+(LV/64)*40+8192: POKE
     LH/7+BV,2   (LH MOD 7)
9330 NEXT LH: RETURN
```

MICRO — 80

Not to worry! The title of this editorial does **not** mean that **MICRO** is going to start covering TRS-80, 8080, or any other processor. **MICRO** is "The 6502 Journal" and has no plans to change that. The title simply refers to 1980 and/or the 1980's. Writing this at the start of a new decade, I want to reflect on what **MICRO** accomplished in the 70's and describe some of its plans for the 80's.

MICRO in the 70's

MICRO was started in 1977 to fill two needs:

1. Provide a quality magazine devoted to the 6502 microprocessor and the various microcomputers based on the 6502. At that time, very little was being printed about the 6502 in the major journals.

2. Provide a means for 6502 oriented dealers and manufacturers to economically reach their specific 6502 audience.

The first issue was printed at a "store front" print shop, ran 28 pages, and had an immediate circulation of 450 copies. Since then **MICRO** has grown in many ways. It is now printed at a commercial printer, is 68 pages or more, has an immediate circulation of almost 10,000 copies, is completely typeset, and is published monthly.

MICRO decided from the start to pay its authors for their material. In fact, we pay twice! Authors received $25.00 per page for material in the magazine, and then received an equal amount for material reprinted in "The BEST of MICRO".

MICRO in the 80's

In the 1980's, we will continue to provide serious articles on 6502 systems, to maintain the Software Catalog, and to continue the on-going 6502 Bibliography. With our monthly format and three week printing/mailing schedule, we will continue to print the most current advertisements.

A number of features will be added. These will include regular "news" columns about each of the major microcomputers; "topical" columns about the use of the 6502 in business, medicine, process control, education, etc.; the **MICROScope** in which qualified reviewers present detailed hardware/software product reviews; a "6502 Club Forum" highlighting club activities; and many other useful features.

To make writing for **MICRO** even more profitable, a new author payment schedule has been established. Authors will now receive up to $50.00 per page for articles as well as residual payments for reprints. The minimum amount per page will be $25.00, with the actual amount dependent on the type of material, quality of the article, etcetera.

I welcome any suggestions you have for improving **MICRO**, and hope that you will continue to participate in the exciting, expanding 6502 world, not just as a **MICRO** reader, but as an active contributor.

Robert M. Tripp

Writing for MICRO

Writing for MICRO is probably easier than you think, and more rewarding too! In this rapidly expanding world of 6502 microcomputers, no single person knows everything, and no single person knows nothing. Every computerist has something to contribute.

MICRO Pays Well

Even though MICRO is much smaller than Kilobaud, Byte, and the other major general microcomputing journals, it pays its authors as much or more than the others in general. Byte, for example, has a published scale of $25 to $50 per page. MICRO pays the same rates. Beyond that, MICRO pays its authors when articles are reprinted in "The BEST of MICRO".This means that a first rate article can earn its author up to $100 per page. If you stop to consider that it normally takes at least three or four pages to present an idea, a discussion and a program, you will realize that it adds up.

MICRO Is Read By 6502 Computerists

Since **MICRO** is totally devoted to the 6502, its readership is composed only of computerists interested in the 6502. Since the general journals cover many different processors, a 6502 article will only appeal to a fraction of the readers, and may easily get lost between TRS-80 junk. An article you write for MICRO will get out to the right people.

MICRO Has Many Opportunities

There are many different ways you can write for MICRO. Each of the ways has its own merit and may apply to you at different times on different topics:

LETTERS and COMMENTS: If you have an observation, suggestion, hint, or other small item of interest which you think others should know about, a 'Letter to the Editor' can be the perfect vehicle. MICRO does not pay for this type of contribution, but you will get full credit with a byline. Small notes about the AIM, SYM, or KIM may be included in "ASK the Doctor", again without payment but with a byline. It doesn't take long to jot down you information and send it in. And, in addition to getting your material in print, you may be really helping other 6502 computerists.

ARTICLES: When you have a larger idea, a complete article is appropriate. While it does take some time and effort on your part to put your information into a form that can be understood by others, it is probably not as difficult as you imagine. The MICRO Staff will work with you to get the article into its final form. You do get paid for any article which is published. While you may never get rich writing articles, you can easily earn enough for that extra memory or whatever.

COLUMNS: We are now actively seeking a few highly qualified individuals to write regular columns. We plan to have a column every other month or so on each of the major 6502 microcomputers, covering news of new products, events, and other items of interest. We also plan to feature regular columns on the use of the 6502 in various fields such as Medicine, Education, Business, Process Control, etcetera, and are looking for writers in these areas. If you are in a position to really know what is happening on one of the 6502 microcomputers or in one of the major application areas, contact us. MICRO will be paying the highest rates for these columns.

MICRO Opportunities

There are numerous opportunities for anyone who wishes to participate in MICRO. We have a Writer's Guide available which will show you in detail how to submit an article to MICRO. Please check the box on the tear-out form in this issue and send it in. MICRO will do the rest.

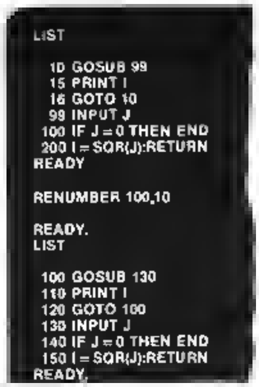
Skyles Electric Works

The BASIC Programmer's Toolkit

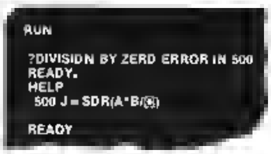
For PET Owners Who Want More Fun And Fewer Errors with Their Programming

Here are Ten Comands you'll need, all on a single chip you can install, in a minute without tools, **on any PET or PET system.** 2 KB of ROM firmware on a single chip with a collection of machine language programs available to you from the time you turn on your PET to the time you shut it off. No tape to load or to interfere with any running programs.

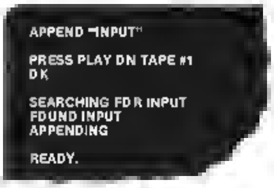
AUTO DELETE RENUMBER HELP TRACE
STEP OFF APPEND DUMP FIND



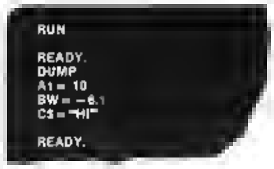
```
LIST

  10 GOSUB 99
  15 PRINT I
  16 GOTO 10
  99 INPUT J
 100 IF J=0 THEN END
 200 I=SQR(J):RETURN
READY

RENUMBER 100,10

READY.
LIST

 100 GOSUB 130
 110 PRINT I
 120 GOTO 100
 130 INPUT J
 140 IF J=0 THEN END
 150 I=SQR(J):RETURN
READY.
```



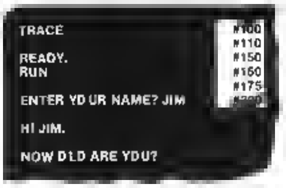
```
RUN

 ?DIVISIDN BY ZERD ERROR IN 500
 READY.
 HELP
  500 J=SDR(A*B/(0))

 READY
```



```
APPEND "INPUT"

PRESS PLAY DN TAPE #1
DK

SEARCHING FDR INPUT
FDUND INPUT
APPENDING

READY.
```



```
RUN

READY.
DUMP
A1= 10
BW= -6.1
C$="HI"

READY.
```



```
TRACE                          #100
                               #110
READY.                         #150
RUN                            #160
                               #175
ENTER YDUR NAME? JIM           #200

HI JIM.

NOW DLD ARE YDU?
```

Can be placed in main board socket or with precision-engineered PCB connector to attach to data bus...depending on the model of your PET and additional memory systems.

Now available to interface

8N/8B, 16N/16B, 32N/32B PET...chip only	**$50.00***
2001-8...chip and interface PCB	**80.00***
With Expandamen, PME 1 **R. C. Factor or Skyles Electric Works systems**	**80.00***
With Computhink Disk System	**85.00***
With Commodore's Word Processor II, for original 2001-8 PETs	**90.00***
With Commodore's Word Processor II, for new PETs	**72.50***
With Skyles Macro TeA	**50.00***

**Shipping and handling, California sales tax where applicable must be added.*

**California residents: please add 6% or 6.5% sales tax as required*

VISA, MASTERCHARGE ORDERS CALL (800) 538-3083 (except California residents)
CALIFORNIA ORDERS PLEASE CALL (408) 257-9140

Skyles Electric Works

10301 Stonydale Drive
Cupertino, California 95014
[408]735-7891

Multiplexing PET's User Port

What do you do when you need to Input or Output more bits of data than your micro can handle? You multiplex! This is not very difficult with a little special hardware and very simple program. This implementation is on a PET, but can be used on any system.

E.D. Morris, Jr.
3200 Washington
Midland, MI 48640

Part of my duties as a chemist involve taking readings from an analytical instrument. The data consists of a series of six digit numbers. These are dutifully copied down on paper and later keypunched into a large computer. The calculations could easily be done in BASIC on a personal computer if there were some way to automatically get the data into the computer.

The data is presented on the front panel as six 7-segment LED readouts. However, the rear panel supplies the data in BCD (Binary Coded Decimal) format. Each decimal digit is represented by four binary bits. Numbers above 9 (binary 1001) are not allowed. For six decimal digits a total of 24 bits is required. Unfortunately most small personal computers such as the PET have only an 8-bit I/O port.

The solution is to multiplex, or combine the data into fewer input lines. For example, each decimal digit has a 1,2,4, and 8 bit. These 24 bits of data could be wired through a 6-position, 4-pole switch to produce four outputs. The computer could then read one digit at a time, change the position of the switch and read again until all six digits are read. The decimal number must then be reconstructed by multiplying each digit by 1, 10, 100, etc., and summing the results.

A mechanical 6-position switch is not really practical for computer operation, but the electronic analog exists in the 74LS151 integrated circuit. The 74LS151 is known as a 1-of-8 data selector and acts like an 8-position single pole switch. This chip has eight inputs (pins 1,2,3,4,12,13,14,15) and one output (pin 5). Three additional pins (9,10,11) control which of the inputs is connected to the output.

If four 74LS151's are used, we have an 8-position, 4-pole switch. The 1's bits from all the decimal digits are connected to one data selector. All of the 2's bits are connected to a second data selector, etc. The output from the four integrated circuits are connected to the four lowest bits (D_0 D_1 D_2 D_3) on the PET input port. The next three bits of the I/O are set to outputs (D_4 D_5 D_6) and used to control the 1-of-8 data selectors. Since I wasn't sure how much current the PET output could supply, I used a 74LS04 hex buffer between the PET outputs and the data selector control lines. The highest bit (D_7) is used as a flag in my application to signal the computer that a number needs to be read.

Figure 1 gives a schematic drawing of the circuit. For clarity, the +5 volt connection (pin 16) and ground connection (pins 7 and 8) are not shown on the data selectors. I built this circuit on a 3" × 4" perf board which plugs directly into the PET user port. If low power logic is used, the circuit requires 5 volts at 20ma. This could be taken from the PET second cassette port. Since Commodore warns against this, I added a 5 volt regulator to my board and stole unregulated 9 volts from the computer. Before plugging this circuit into your computer, you should power it up with an external supply and verify that each input works when tested with a voltmeter.

The following program will allow the PET to read a 6-digit decimal number through the user port.

```
10 POKE 59459,112
20 A=59471

30 FOR I=0 TO 5

40 P=I*16

50 POKE A,P

60 B(I)=PEEK(A)AND15

70 NEXTI

80 C=B(0)+10*B(1)+100*B(2)·
 +1000*B(3)+10000*B(4)
 +100000*B(5)

90 PRINT C
```

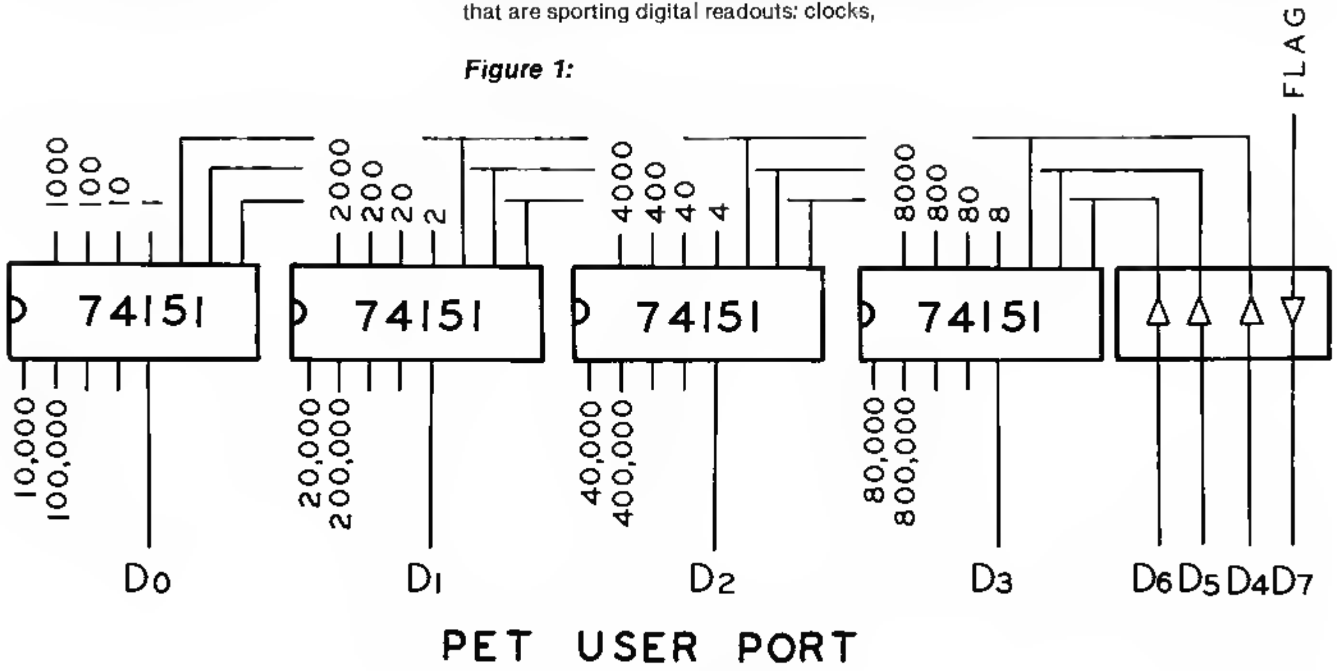
Explanation of Program

Line 10 Sets up D4 D5 and D6 as outputs
Line 20 User Port address
Line 50 Sends signal to data selectors
Line 60 Reads lower four bits & masks out others
Line 80 Reconstructs decimal number from digits
Line 30 It I goes from 0 to 7, then all 32 bits are read.

I am using only 24 bits, however, the circuit described here will read up to 32 bits through an 8-bit I/O port. It you don't need D7 tor a flag, you can use the 74LS150 1-ot-16 data selector to read 64 bits. D7 would then be a fourth control line.

You probably don't have an analytical instrument around the house to keep track of, but look at all the other devices that are sporting digital readouts: clocks, timers, scanners, thermometers, TV channel selectors, etc. The data for these is normally generated in BCD tormat and then converted to 7-segment tor display. A multiplexing technique can be used whenever you have more bits of data than input ports. The bits don't have to be a decimal number; each bit could represent ot sensor of a burglar alarm system or the position of a turnout in a model train layout.

Figure 1:



The Binary Sort

Here is a concise description of the Binary Sort concept, and a detailed implementation in BASIC that should be easy to adapt to any micro or application.

Robert Phillips
6 McKee Avenue
Oxford, OH 45056

Sometimes we have an array of data which we need to search in order to find the location of one particular element in it. This is more common with alphabetic data, but we may have to do it with either alpha or numeric data. The simplest way to find the item is to use a FOR-loop, checking each item individually until we find the one we are looking for. The average number of steps through the loop that must be made to find a given item is approximately half the length of the list. If the item is not on the list, then the program must execute as many steps through as there are items on the list. When the array is short, there is no problem. However, as the array gets longer, this method becomes more and more inefficient. An array that has 500 elements in it will require an average of 250 steps through the loop to find an item. Such a search will take several seconds.

When the list is ordered (i.e., sorted into either ascending or descending order), there is a much more efficient way to search the list: the binary search. Basically stated, in a binary search you continually divide the list into two halves and then eliminate the half which cannot contain your item. (Because the list is always divided into two halves, this is called a binary search.) For example, if the item at the half-way point is larger than the item you are looking for, you know that your item cannot be in the second half of the list. So, you eliminate it from consideration. You then divide the remaining list in half, and continue the process of eliminating and dividing until you find the item, or until you cannot cut in half any more. If that happens, the item you are looking for is not on the list, and your search has failed.

In a FOR-loop search, each step through the loop elimates only one item from the list; in a binary search, each step through eliminates half of the remaining list. Taking as an example a list of 255 items, Table 1 shows how much is eliminated at each iteration through the loop. The first column is the step number, the second column gives how many were eliminated in that step, and the third tells the total number of items now eliminated.

After step 8 through the search, you have either found your item (and you may well have found it before step 8), or your search has failed. At any rate, it took you only 8 times through the loop to find your item, as opposed to the average of 128 (maximum: 255) that a straight search would require. The best part is that if you double the list, the binary search requires only one more step through the loop; double it again, and add just one more time through! Obviously, this is a wonderful tool.

There are only two requirements for a binary search: 1) the list must be in order; and 2) the items on the list must be unique (or, if not, it doesn't matter to you which of the duplicated items is located).

To do a binary search, we need two variables. One to point at where we are in the array, and one to keep cutting the search-field in half. In Table 2, I call them

Step No.	Eliminated this step	Total eliminated
1	128	128
2	64	192
3	32	224
4	16	240
5	8	248
6	4	252
7	2	254
8	1	255

Table 1.

Step	PT	IV	Find?	New IV	+ *or* −	New PT
1	8	8	no	4	+	12
2	12	4	no	2	−	10
3	10	2	no	1	+	11
4	11	1	YES!			

Table 2.

PT (for "pointer") and IV (for "interval"). IV will get cut in halt each time through, until it gets down to 1. IV will be added to PT if we have to go further down the list; it will be subtracted from PT it we have to come back up higher on the list. To illustrate this, let us assume an array of 15 elements. The item we are searching tor happens to be in position 11. Let's step through and see what happens to PT and IV at each step.

The logic to do this is not difficult. Let's say that our array is called L1$, and is an alpha array sorted into ascending (i.e., alphabetical) order. We have another variable TL ("total" — it is the same variable we would have used in a FOR-loop: FOR I 1 to TL) which tells us how many items are currently in the array. Finally, the item we are trying to find is stored in the variable SW$. The simple algorithm appears in Figure 1.

It the array were sorted into descending order, the " " and " " symbols in statements 40 and 50 would be reversed. Notice that we use the INT function and round up. This is the equivalent to the CEILING function. Both things are necessary; if you don't round up, you won't be able to get to the end ot the list, and non-integers will get clobbered during the division process.

As it happens, I do not like the redundancy of lines 40 and 50; I preter to make them a little more etticient. I do it so that IV is always added to PT. Then, with one compare, I find out if IV should be positive (so that the addition will add IV to PT) or negative (so that the addition will, in effect, subtract IV from PT). So, I prefer to have lines 40 and 50 as follows:

```
40 IF L1$ (PT)  SW$  THEN IV = ·IV

50 PT =  PT + IV
```

While this is certainly more "elegant," It also adds a problem. IV will quite often turn out negative, and that will really toul up what happens In statement 30. So, we have to change 30 to:

```
30 IV  =  INT((ABS(IV))/2 + .5.
```

```
10  PT=INT(TL/2+.5): IV=PT

20  IF L1$(PT)=SW$ THEN GOTO [you have
  found it!]

30   IV=INT(IV/2+.5)

40  IF L1$(PT) SW$ THEN IV=IV-PT

50  IF L1$(PT) SW$ THEN IV=IV+PT

60  GO TO 20
```

Figure 1.

Now, having added the ABS tunction into line 30 to ensure that IV will always be positive, I am not sure that I have gained anything in efficiency. But, I think that it is more elegant, so I'll leave it!

If you try to run the program the way it is, you may have a problem: If the Item that you are searching tor is not on the list, you will get into an infinite loop and the only way out of the algorithm is to find the item. So, we have to check to see if IV has the value ot 1. It it does we cannot cut In half any more; we cannot search any more. We need to test IV's absolute value, and I put it right after the compare, calling it line 25.

```
25 IF  ABS(IV) = 1  THEN  GOTO
[the search has failed]
```

If everything in the world were perfect, that would be the algorithm. However, since consistently rounding IV up for the reasons pointed out above, we may actually, at some times, exceed the bounds ot the array, raising the error condition. There are several different ways to handle the problem; I believe the easiest is to take the value of IV away from PT and continue on from there. Since I don't know at this point It IV is negative or positive, I simply change its sign and add it to PT in line 55.

```
55 IF PT    TL OR PT    1
THEN IV = – IV: PT = PT + IV
```

(If you really don't like to have IV go negative and then to have to use ABS, you can use the original version ot lines 40 and 50, and then use two statements here In place of 55.

```
IF PT    1 THEN PT = PT + IV
and IF PT    TL THEN PT = PT – IV)
```

My version of the binary sort algorithm is shown in Figure 2.

```
10  PT=INT(TL/2+.5): IV=PT

20  IF L1$(PT)=SW$ THEN GOTO [found it! PT
  is the number of the item]

25  IF ABS (IV)=1 THEN GOTO [the search
  has apparently failed]

30  IV=(INT((ABS(IV))/2+.5)

40  IF L1$(PT)  SW$ THEN IV=-IV

50  PT=PT+IV

55  IF PT  TL OR PT  1 THEN IV=-IV: PT=PT+IV

60  GOTO 20
```

Figure 2

There is, unfortunately, still one more potential problem. If the number of items in the array (TL) is exactly a power of 2 (16, 32, 64, 128,etc.), the search will not locate the very last item In the array. The reason is that when you cut in half, you don't cut perfectly in halt. If the array has 16 elements in it, you look first at element 8: there are actually 7 elements above it in the array; but there are 8 elements below it! If the array has any number other than a power ot 2, there is always one division which has to be rounded up, and that rounding up gives us room to get to the very end of the array. (Actually, it also caused the problem ot going beyond the bounds of the array, which made us add line 55.) There are several ways to overcome the problem, including preventing the array ever from having an "undesirable" number of items. For me, the simplest thing to do is to check the last item in the array it the search fails. If they don't match, then the search actually has failed. But If it does succeed at this point, I do have to assign the value of TL to PT, as PT is what is carried into the main program to tell what item number was found. I do the entire thing in line 70:

```
70 IF SW$ = L1$(TL)
THEN PT = TL: GOTO [tound it!]
```

I also have to change line 25, so that the GOTO there branches to 70.

If the compare in line 70 yields a false, then the search has really failed, and you drop out ot the binary search algorithm. Let's now look at the complete algorithm in Figure 3, which is missing only the line numbers after the GOTO statements which will link the search to the programs you use it in.

```
10  PT=INT (TL/2+.5): IV=PT

20  IF L1$(PT)=SW$ THEN GOTO [found it]

25  IF ABS (IV)=1 THEN GOTO 70

30  IV=INT ((ABS(IV))/2+.5)

40  IF L1$(PT)  SW$ THEN IV=-IV

50  PT=PT+IV

55  IF PT  TL OR PT  1 THEN IV=-IV: PT=PT+IV

60  GOTO 20

70  IF  SW$=L1$(TL)  THEN  PT=TL:  GOTO
  [found it]

80  REM Search has failed and you're out
  of the binary search algorithm.
```

Figure 3

Announcing...

OPTIMIZED SYSTEMS SOFTWARE
UPGRADE YOUR APPLE II® WITH A NEW SYSTEMS SOFTWARE PACKAGE

- Unified Operating System
- Disk File Manager
- Commercial Basic
- Editor/Assembler/Debugger
- Data Base Manager

Optimized Systems Software does not use Apple DOS®. OSS is a unified and complete systems software package with its own Operating System and File Manager. The Operating System, the File Manager and the Basic combined use only slightly more RAM than Apple DOS® alone. Requires 48K Apple II® with Disk II.

Operating System

- Byte and Block I/O
- Simple User Interface
- Simple Device Interface
  (create your own)

Basic

- Nine Digit Precision DECIMAL Floating Point
- 32K Byte Strings
- Variable Names to 256 significant characters
- I/O Interface Statements
  (no PRINT "control-D...")

File Manager

- Open, Read, Write, Delete, Lock, etc.
- Random Access via Note & Point
- File Names of Primary.Ext type

Editor/Assembler/Debugger

- Line Editor
  (Edits Basic programs, too)
- Mini Assembler
- Maxi Assembler
- Disassembler
- Step, Trace, etc.

Available **NOW at Special Introductory Prices**

● Operating System + File Manager	$24.95
● Operating System + File Manager + Basic	$49.95
● Operating System + File Manager + ASM	$49.95
● Operating System + File Manager + Basic + ASM	$89.95
● Operating System + Data Base Manager	(2nd O)

Order today. Add $2.00 for shipping & handling. California residents add 6% sales tax. Visa/Mastercharge welcome. Personal checks require 2 weeks to clear.

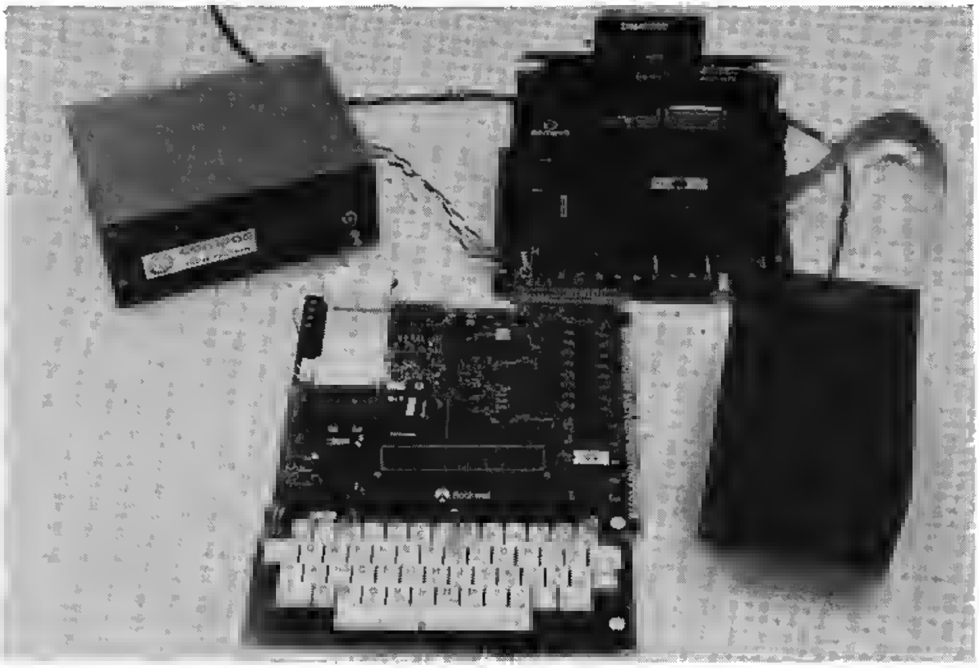
Note: Apple II®, Apple DOS® are trademarks of Apple Computer, Inc.

Optimized Systems Software
Shepardson Microsystems, Inc.
20823 Stevens Creek Blvd., Bldg. C4-H
Cupertino, CA 95014
(408) 257-9900

compas
microsystems

P.O. Box 687
224 S.E. 16th Street
Ames, Iowa 50010

DAIM



DAIM is a complete disk operating system for the ROCKWELL INTERNATIONAL AIM 65. The DAIM system includes a controller board (with 3.3K operating system in EPROM) which plugs into the ROCKWELL expansion motherboard, packaged power supply capable of driving two 5 1/4 inch floppy drives and one or two disk drives mounted in a unique, smoked plastic enclosure. DAIM is completely compatible in both disk format and operating system functions with the SYSTEM 65. Commands are provided to load/save source and object files, initialize a disk, list a file, list a disk directory, rename files, delete and recover files and compress a disk to recover unused space. Everything is complete — plug it in and you're ready to go! DAIM provides the ideal way to turn your AIM 65 into a complete 6500 development system. Also pictured are CSB 20 (EPROM/RAM) and CSB 10 (EPROM programmer) which may be used in conjunction with the DAIM to provide enhanced functional capability. Base price of $850 includes controller board with all software in EPROM, power supply and one disk drive. Now you know why we say —

There is nothing like a

DAIM

Phone 515-232-8187

A Complete Morse Code Send/Receive Package for the AIM 65

Here is a valuable program for any AIM user. While it will be of most interest to a HAM radio buff, the techniques which include the use of timers, interrupts, table lookups, and so forth should be instructive to everyone.

Marvin L. DeJong
Dept. of Math & Physics
The School of the Ozarks
Pt. Lookout, MO 65726

I. FEATURES:

A . Send Morse code using the AIM 65 keyboard. A 256 character buffer permits typing ahead.

B . Send pre-loaded Morse code messages. Three messages totaling 256 characters can be sent.

C . A simple interface circuit allows the program to operate as an electronic keyer.

D . Code speed in words per minute is entered on the keyboard and displayed on the AIM 65 display

E . Control of the entire program is from the keyboard.

E . A single integrated circuit provides the interface for receiving Morse code.

G . The received code is converted to alphanumeric characters on the AIM 65 display, and is scrolled left as the code is received.

H . Code speed is adjustable from 5 to 99 wpm.

II. OPERATING INSTRUCTIONS

The following paragraphs serve as an operating guide for the program.

A . Load the program given in the listings and construct the interface circuits shown in Figures 1 and 2. The cross-coupled NAND gate interface in Figure 1 is not needed if you do not operate the program as a paddle-type electronic keyer. Set the P register to zero before starting the program.

B . Execution begins at address $0500. After initializing the program, three messages (called A, B and C) may be entered from the AIM 65 keyboard. As messages are entered they will appear on the display, and they will be recorded by the thermal printer if the printer is on. If a mistake is made, pressing the DEL key will clear the character and a new character may be entered. The RETURN key is pressed when a message is complete. An example of a message is " CQ CQ CQ DE KOEI KOEI K." Message A is the first one entered, message C is the last. The sum of the characters including spaces cannot exceed 256. Pressing the RETURN key at the end of the third message causes the program to proceed to the keyboard-send mode. If you do not have any messages to place in memory, hit the space bar and the RETURN key three times in succession to enter the keyboard-send mode.

C . In the keyboard-send mode, pressing a key will cause the corresponding Morse character to be sent, while pressing a control key will cause the corresponding control operation (described below) to be carried out. The keyer will also operate at this time if you wish to use the keyer rather than the keyboard.

D . The first thing you will want to do in the keyboard-send mode is set the code speed. Press the CTRL key; and , while holding down the CTRL key, press the S key (S is for "speed"). Release these keys and then enter the code speed at which you wish to operate. The two-digit decimal number should appear at the far left of the display.

E . Pressing CTRL A, B, or C will cause the corresponding message to be sent. Any set of spaces in any of the messages may be interrupted by the keyer (to fill in an RST report, for example), but they will not be interrupted by keyboard entries other than control functions.

F . Morse code may be sent from the keyboard by typing the characters. They appear on the display as they are typed, and they disappear from the display when they are sent. You can type ahead of the Morse code being sent by filling a 256 character buffer. (No warning is given for a full buffer because, in my experience, you rarely get 256 characters ahead.) If while sending Morse code with the keyboard you find that you have made a mistake, perish the thought, a delete function has thoughtfully been provided. Use the DEL key to try to get to the mistake before the send program gets to the character (this can be challenging at high code speeds or with slow fingers). Also, if you delete when there are no characters left to delete, you will get the contents of the entire buffer. Hit the RETURN key if this happens. RETURN starts the entire program over.

G . The RETURN key serves as a panic button. It will restart the program when you are in the keyboard-mode. It can get you out of desperate situations. The RETURN key followed by the F1 key puts you right back in the keyboard-send mode without affecting the messages A, B, and C.

H . The speed can be changed at any time, even in the middle of a message or when the send buffer has characters left to be sent. However, the CTRL S interrupts the program until the two-digit number is entered; so if you are in the middle of a dot or dash, the transmitter will remain on until you finish entering the speed. At that time the code element, the character, and the remaining message will be sent at the new speed.

I . If you wish to preload the buffer while the "other guy" is sending, you can press CTRL L (L is for "load"). The program loops while you load the buffer.

J . CTRL K returns the program from the load loop (or the receive mode) to start sending the code in the buffer. CTRL K always sends the program back to the keyboard-send mode, disabling the CTRL L mode and the receive mode.

K . CTRL R sends the program to receive code. The program will copy code over a wide range of code speeds, so adjustments in the code speed are infrequent. However, if you want to be "right on," the left-most digit of the speed display will blink if your speed is too fast, while the right-most digit will blink if your speed is too slow. Blinking digits are produced by measuring the incoming dot length. Variations in the dot length of the incoming code may cause both digits to blink. Then you are "right on!" Noise spikes are typically regarded as excessively short dots and will cause the left-most digit to blink.

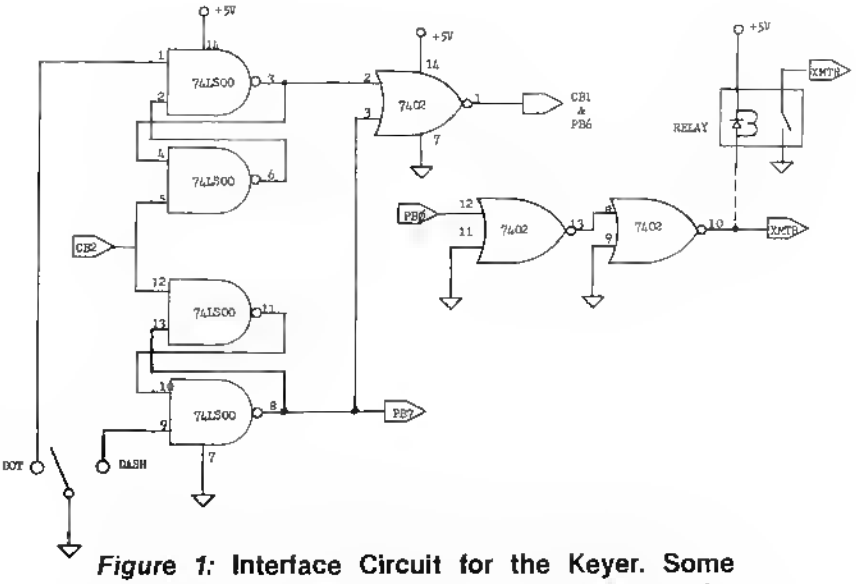


Figure 1: **Interface Circuit for the Keyer. Some transmitters will require a relay for keying. This interface circuit may be omitted if you do not wish to operate in the keyer mode.**

L Do not spend a lot of time trying to zero-in on someone's code speed. The finite resolution of the speed settings prevent a measurement that is more accurate than about 2 wpm. Variations in the weight ratio and other personal characteristics of sending will also affect the actual speed. The code-speed measurement will be accurate for machine-sent code, from W1AW or another AIM 65 for example. The received code will appear on the AIM 65 display moving from right to left. A too-high speed setting is better than too low.

M . The bandwidth of the interface circuit, an LM567 tone decoder, is narrow, so tuning is delicate. Watch the LED output carefully until it blinks in syncronism with the incoming code. Practice copying W1AW broadcasts until you become familiar with the operating of the receive mode. Remember that an AIM 65 and an LM567 are somewhat less powerful than the human mind and the ear when copying faint signals in the presence of noise.

N . You can return from the receive mode to the keyboard-send mode by the CTRL K operation.

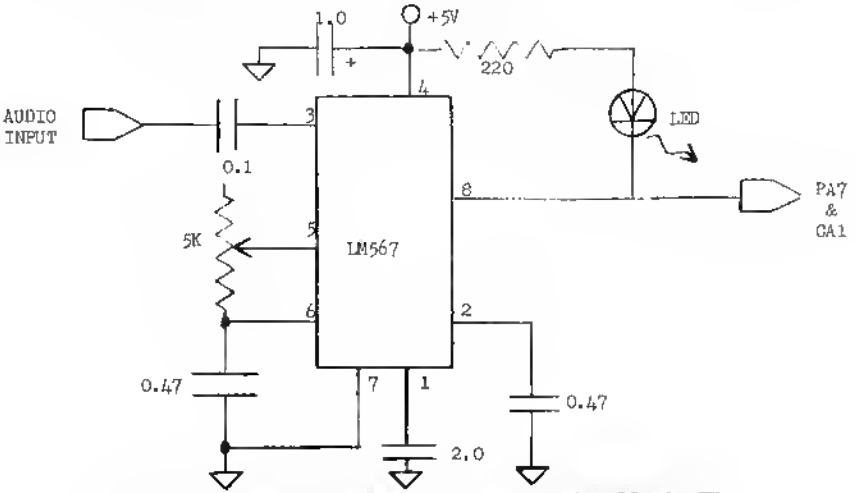


Figure 2: **Interface Circuit for the Receive Mode. The 5K potentiometer is adjusted to correspond to the center frequency of the CW note. The signal is tuned with the receiver until the LED flashes in unison with the code being received.**

TABLE I. Routine Location Table.

LOCATIONS		FUNCTION
$0200 – $02FF	–	Messages A, B, and C are stored in these locations.
$0300 – $03FF	–	Keyboard buffer. Holds up to 256 characters so you can type ahead.
$0420 – $045C	–	ASCII to Morse Code Conversion Table
$0480 – $04D7	–	Morse Code to ASCII Conversion Table
$04F3	–	Conversion of comma (,) in Morse Code to ASCII.
$0500 – $0564	–	Routine to initialize certain registers and input the three messages with the keyboard.
$0565 – $0582	–	Set up interrupt vector and start servicing the keyboard on an interrupt basis.
$0583 – $058E	–	Initialize the keyboard buffer memory locations.
$058F – $05A2 and $05F4 – $05F9	–	Keyboard wait loop. Program waits here until a keyboard entry has been made to the buffer. When such an entry is made, the program sends the character.
$05A3 – $05F3	–	Subroutine SEND. Contains subroutine DOT at $05CB, subroutine DASH at $05E4, and subroutine TIMER at $05E9.
$0600 – $065F and $09A7 – $09C7	–	Subroutine KEYBOARD. This subroutine is part of the interrupt routine that scans the keyboard. If a key has been depressed, it stores the ASCII character in the buffer, unless it is a control character. If it is a control character, the appropriate control function is implemented. For example, Control R sends the program to the receive routine.
$0660 – $0671	–	Subroutine DISPLAY. Used to display characters on the AIM 65 display.
$0672 – $0684	–	Subroutine MODIFY. Used to shift the elements in the display buffer to the left.
$0685 – $069A	–	Subroutine BACKSPACE. Used to shift the elements in the display buffer to the right, entering a blank (space) for a deleted character.
$069B – $06A5	–	Subroutine CLEAR. Used to clear the display buffer.
$06A6 – $06BF	–	Subroutine NONAME. Used to clear the display location that contained the character just converted to Morse code.
$06C0 – $06E5	–	Interrupt routine for keyer.
$06E6 – $06ED and $0904 – $09A6	–	Interrupt routine to scan the keyboard.

III. BACKGROUND

Morse code send/receive programs have appeared in several forms in the literature. Consult the bibliography for some useful references. The routines used in this program have previously been described by the author's articles in MICRO (MICRO is published by MICRO INK, Inc., P.O. Box 6502, Chelmsford, MA 01824), and will not be described in detail here. Table 1 locates the various routines, and the references given in the bibliography will explain most of these routines.

The keyboard is read on an interrupt basis, making extensive use of the monitor subroutine ONEKEY at $ED05. Also, the keyboard-read routine duplicates the monitor subroutine GETKEY at $EC40, with some important modifications for interrupt operation. The T1 timer on the user 6522 is used to produce interrupts every $8000 microseconds, at which time the keyboard is scanned.

The Morse code receive algorithm may be summarized as follows: Define the presence of a tone as a *mark* and the absence of a tone as a *space*. The receive program idles in a loop until the leading edge of a mark element produces an interrupt request (IRQ). At that time, a mark-counter memory location is incremented at 1024 microsecond intervals until the mark is gone. During a space a space-counter memory location is incremented. When the space-counter is equal to ½ the dot length as determined by the speed setting, then the mark-counter memory location is examined to determine if the mark was a noise pulse, a dot, or a dash. If the mark counter was less than ½ the dot length, the mark is regarded as a noise pulse. If the mark counter is between ½ the dot length and twice the dot length, the mark is regarded as a dot. If the mark counter exceeds twice the dot length, the mark is recorded as a dash.

As soon as a decision is made about the mark counter, it is cleared to prepare it for the reception of the next Morse code element. Meanwhile, the space counter is continually being incremented once every 1024 microseconds. When it exceeds twice the dot length, the program concludes that an entire Morse character has been received; and the corresponding alphanumeric character is displayed on the AIM 65 display. As the space counter is incremented further, it reaches four times the dot length; at which time the program decides that a word space has been sent, and a space appears on the AIM 65 display. At this time the space counter is cleared, the speed setting is checked to see if the operator changed the speed setting on the AIM 65, and the program returns to the wait loop to wait for the next mark.

The author is aware of receive programs that use automatic calibration of tracking on the incoming code speed. Consult the bibliography for details. My own experience is one of frustration because the presence of noise and interfering signals affects the automatic calibration, although I have heard reports that Bob Kurtz's program works nicely. In the present case, we have used manual control of the code speed with good results. Some experience and practice is useful. Bob Kurtz's program could be adapted for the AIM 65, and could also be adapted to work with the present send programs.

TABLE I. Routine Location Table, continued.

LOCATIONS		FUNCTION
$06EE – $073F	–	Interrupt routine for Morse code receive program.
$0750 – $07A5	–	Control S routine. Converts decimal entry of speed to the number needed to load the timer.
$07AB – $07B5	–	Subroutine TMELOAD. Used to load the timer for the receive program.
$07B6 – $07C3	–	Subroutine UNTITLED. Used to display the Morse code character that has just been decoded by the receive program.
$0820 – $0901	–	Receive routine.

IV. BIBLIOGRAPHY

1. Pollock, James W., "1000 WPM Morse Code Typer," 73, January 1977, p. 100.
2. Ockers, Stan, "Code Test," *The First Book of Kim*, Orb, Argonne, Illinois, 1977, p. 56.
3. Pollock, James W., "A Microprocessor Controlled CW Keyboard," Ham Radio, January 1978, p. 81
4. DeJong, Marvin L., "A Complete Morse Code Send/Receive Program for the KIM-1," MICRO, April-May 1978, p. 7.
5. Kurtz, Bob, "Morse Code Reader Program," 6502 User Notes, No. 11, p. 9.
6. DeJong, Marvin L., "Build the KIM Keyer," 73, September 1979, p. 80.
7. DeJong, Marvin L., "An AIM 65 Notepad," MICRO, September 1979, p. 11.
8. DeJong, Marvin L., "AIM 65 in the Ham Shack," MICRO, September 1979, p. 29.

Morse Code Listings

```
<M>=0420 00 00 00 00
< > 0424 00 00 00 CE
< > 0428 00 00 00 00
< > 042C CE 8C 56 94
< > 0430 FC 7C 3C 1C
< > 0434 0C 04 84 C4
< > 0438 E4 F4 16 32
< > 043C 00 8C 00 32
< > 0440 00 60 88 A8
< > 0444 90 40 28 D0
< > 0448 08 20 78 B0
< > 044C 48 E0 A0 F0
< > 0450 60 D0 50 10
< > 0454 C0 30 10 70
  > 0458 98 B8 C8 00
< > 045C 20 20 20 20
```

```
<M>=0480 20 20 45 54
< > 0484 49 41 4E 4D
< > 0488 53 55 52 57
< > 048C 44 4B 47 4F
  > 0490 48 56 46 20
< > 0494 4C 20 50 4A
< > 0498 42 58 43 59
< > 049C 5A 51 20 20
< > 04A0 35 34 20 33
< > 04A4 20 20 20 32
< > 04A8 20 20 20 30
< > 04AC 20 20 20 31
< > 04B0 36 3D 2F 20
< > 04B4 20 20 20 20
< > 04B8 37 20 20 20
< > 04BC 38 20 39 30
< > 04C0 20 20 20 20
  > 04C4 20 20 20 20
< > 04C8 20 20 20 20
< > 04CC 3F 20 20 20
<M>=04F0 20 20 20 2C
<K>*=0500
/50
 0500 78 SEI
 0501 A9 LDA #A0
 0503 8D STA A00C
 0506 A9 LDA #01
 0508 8D STA A000
 050B 8D STA A002
 050E 20 JSR 069B
 0511 A2 LDX #00
 0513 A0 LDY #00
 0515 94 STY 01,X
 0517 20 JSR E93C
 051A 20 JSR F000
 051D C9 CMP #7F
 051F D0 BNE 0531
 0521 A9 LDA #20
 0523 88 DEY
 0524 99 STA 0200,Y
 0527 8A TXA
 0528 48 PHA
```

```
0529 20 JSR 0685
052C 68 PLA
052D AA TAX
052E 18 CLC
052F 90 BCC 0517
0531 C9 CMP #5E
0533 F0 BEQ 055F
0535 C9 CMP #0D
0537 F0 BEQ 054C
0539 99 STA 0200,Y
053C 8A TXA
053D 48 PHA
053E B9 LDA 0200,Y
0541 20 JSR 0672
0544 20 JSR 0560
0547 68 PLA
0548 AA TAX
0549 C8 INY
054A D0 BNE 0517
054C 8A TXA
054D 48 PHA
054E 20 JSR 069B
0551 20 JSR 0666
0554 68 PLA
0555 AA TAX
0556 88 DEY
0557 94 STY 04,X
0559 C8 INY
055A E8 INX
055B E0 CPX #03
055D 90 BCC 0515
```

Morse Code Listings, cont'd.

```
<K>*=055F
/50
 055F 20 JSR 069B
 0562 20 JSR 0660
 0565 A9 LDA #C0
 0567 8D STA A404
 056A A9 LDA #06
 056C 8D STA A405
 056F A9 LDA #D0
 0571 8D STA A00E
 0574 A9 LDA #40
 0576 8D STA A00B
 0579 A9 LDA #FF
 057B 8D STA A006
 057E A9 LDA #FF
 0580 8D STA A005
 0583 A9 LDA #00
 0585 85 STA 20
 0587 85 STA 22
 0589 A9 LDA #03
 058B 85 STA 21
 058D 85 STA 23
 058F A0 LDY #00
 0591 58 CLI
 0592 A5 LDA 22
 0594 C5 CMP 20
 0596 F0 BEQ 0592
 0598 B1 LDA (20),Y
 059A AA TAX
 059B 20 JSR 05A3
 059E E6 INC 20
 05A0 4C JMP 05F4
 05A3 BD LDA 0400,X
 05A6 F0 BEQ 05C6
 05A8 0A ASL  A
 05A9 F0 BEQ 05BB
 05AB 48 PHA
 05AC B0 BCS 05B4
 05AE 20 JSR 05CB
 05B1 4C JMP 05B7
 05B4 20 JSR 05E4
 05B7 68 PLA
 05B8 4C JMP 05A8
 05BB A2 LDX #02
 05BD 20 JSR 05E9
 05C0 CA DEX
 05C1 D0 BNE 05BD
 05C3 60 RTS
 05C4 D8 CLD
 05C5 EA NOP
 05C6 A2 LDX #04
 05C8 4C JMP 05BD
```

```
<K>*=05CB
/50
 05CB A2 LDX #01
 05CD CE DEC A000
 05D0 20 JSR 05E9
 05D3 CA DEX
 05D4 D0 BNE 05D0
 05D6 AD LDA A000
 05D9 4A LSR  A
 05DA B0 BCS 05E3
 05DC EE INC A000
 05DF E8 INX
 05E0 4C JMP 05D0
 05E3 60 RTS
 05E4 A2 LDX #03
 05E6 4C JMP 05CD
 05E9 A5 LDA 07
 05EB 8D STA A497
 05EE 2C BIT A497
 05F1 10 BPL 05EE
 05F3 60 RTS
 05F4 20 JSR 06A6
 05F7 4C JMP 0592
 05FA EA NOP
 05FB EA NOP
 05FC EA NOP
 05FD EA NOP
 05FE EA NOP
 05FF EA NOP
 0600 48 PHA
 0601 29 AND #E0
 0603 F0 BEQ 0623
 0605 68 PLA
 0606 C9 CMP #7F
 0608 D0 BNE 0612
 060A C6 DEC 22
 060C D8 CLD
 060D EA NOP
 060E 20 JSR 0685
 0611 60 RTS
 0612 C9 CMP #5B
 0614 B0 BCS 0611
 0616 A0 LDY #00
 0618 91 STA (22),Y
 061A E6 INC 22
 061C 20 JSR 0672
 061F 20 JSR 0660
 0622 60 RTS
 0623 68 PLA
 0624 C9 CMP #13
 0625 D0 BNE 062B
 0628 4C JMP 0750
```

```
<K>*=062B
/50
 062B C9 CMP #04
 062D B0 BCS 0649
 062F A8 TAY
 0630 B6 LDX 00,Y
 0632 8A TXA
 0633 48 PHA
 0634 BD LDA 0200,X
 0637 AA TAX
 0638 58 CLI
 0639 20 JSR 05A3
 063C 78 SEI
 063D 68 PLA
 063E AA TAX
 063F D9 CMP 0003,Y
 0642 F0 BEQ 0648
 0644 E8 INX
 0645 4C JMP 0632
 0648 60 RTS
 0649 C9 CMP #0D
 064B D0 BNE 0650
 064D 4C JMP 0500
 0650 C9 CMP #12
 0652 D0 BNE 0657
 0654 4C JMP 0820
 0657 4C JMP 09A7
 065A EA NOP
 065B EA NOP
 065C EA NOP
 065D EA NOP
 065E EA NOP
 065F EA NOP
 0660 A2 LDX #13
 0662 8A TXA
 0663 48 PHA
 0664 BD LDA A438,X
 0667 09 ORA #80
 0669 20 JSR EF7B
 066C 68 PLA
 066D AA TAX
 066E CA DEX
 066F 10 BPL 0662
 0671 60 RTS
 0672 8D STA A44C
 0675 A2 LDX #03
 0677 BD LDA A439,X
 067A CA DEX
 067B 9D STA A438,X
 067E E8 INX
 067F E8 INX
 0680 E0 CPX #15
```

Morse Code Listings, cont'd.

```
<K>*=0682
/50
 0682 90 BCC 0677
 0684 60 RTS
 0685 A2 LDX #10
 0687 BD LDA A43A,X
 068A E8 INX
 068B 9D STA A43A,X
 068E CA DEX
 068F CA DEX
 0690 10 BPL 0687
 0692 A9 LDA #20
 0694 8D STA A43A
 0697 20 JSR 0660
 069A 60 RTS
 069B A2 LDX #13
 069D A9 LDA #20
 069F 9D STA A438,X
 06A2 CA DEX
 06A3 10 BPL 069F
 06A5 60 RTS
 06A6 38 SEC
 06A7 A5 LDA 22
 06A9 E5 SBC 20
 06AB C9 CMP #12
 06AD B0 BCS 06BF
 06AF 85 STA 24
 06B1 38 SEC
 06B2 A9 LDA #11
 06B4 E5 SBC 24
 06B6 AA TAX
 06B7 A9 LDA #20
 06B9 9D STA A43A,X
 06BC 20 JSR 0660
 06BF 60 RTS
 06C0 48 PHA
 06C1 8A TXA
 06C2 48 PHA
 06C3 98 TYA
 06C4 48 PHA
 06C5 AD LDA A00D
 06C8 29 AND #10
 06CA F0 BEQ 06E6
 06CC AD LDA A000
 06CF 30 BMI 06D7
 06D1 20 JSR 05CB
 06D4 4C JMP 06DA
 06D7 20 JSR 05E4
 06DA AD LDA A000
 06DD 0A ASL .A
 06DE 10 BPL 06DC
 06E0 68 PLA
```

```
<K>*=06E1
/50
 06E1 A8 TAY
 06E2 68 PLA
 06E3 AA TAX
 06E4 68 PLA
 06E5 40 RTI
 06E6 2C BIT A00D
 06E9 50 BVC 06EE
 06EB 4C JMP 0904
 06EE 20 JSR 07AB
 06F1 A9 LDA #20
 06F3 2C BIT A00D
 06F6 F0 BEQ 06F3
 06F8 AD LDA A001
 06FB 10 BPL 0710
 06FD E6 INC 1A
 06FF D0 BNE 0703
 0701 E6 INC 1B
 0703 EA NOP
 0704 EA NOP
 0705 EA NOP
 0706 EA NOP
 0707 EA NOP
 0708 EA NOP
 0709 68 PLA
 070A A8 TAY
 070B 68 PLA
 070C AA TAX
 070D 68 PLA
 070E 40 RTI
 070F EA NOP
 0710 20 JSR 07AB
 0713 E6 INC 18
 0715 D0 BNE 0719
 0717 E6 INC 19
 0719 A5 LDA 19
 071B C5 CMP 1D
 071D 90 BCC 06F1
 071F D0 BNE 0727
 0721 A5 LDA 18
 0723 C5 CMP 1C
 0725 90 BCC 06F1
 0727 A9 LDA #00
 0729 85 STA 1A
 072B 85 STA 1B
 072D A9 LDA #20
 072F 2C BIT A00D
 0732 F0 BEQ 072F
 0734 AD LDA A001
 0737 10 BPL 0710
 0739 AD LDA A001
```

```
<K>*=073C
/50
 073C 58 CLI
 073D 4C JMP 085E
 0740 EA NOP
 0741 EA NOP
 0742 EA NOP
 0743 EA NOP
 0744 EA NOP
 0745 EA NOP
 0746 EA NOP
 0747 EA NOP
 0748 EA NOP
 0749 EA NOP
 074A EA NOP
 074B EA NOP
 074C EA NOP
 074D EA NOP
 074E EA NOP
 074F EA NOP
 0750 20 JSR E93C
 0753 48 PHA
 0754 8D STA A438
 0757 20 JSR 0660
 075A 68 PLA
 075B 38 SEC
 075C E9 SBC #30
 075E 0A ASL .A
 075F 0A ASL .A
 0760 0A ASL .A
 0761 0A ASL .A
 0762 85 STA 11
 0764 20 JSR E93C
 0767 48 PHA
 0768 8D STA A439
 076B 20 JSR 0660
 076E 68 PLA
 076F 38 SEC
 0770 E9 SBC #30
 0772 18 CLC
 0773 65 ADC 11
 0775 48 PHA
 0776 29 AND #F0
 0778 4A LSR  A
 0779 85 STA 10
 077B 4A LSR .A
 077C 4A LSR .A
 077D 18 CLC
 077E 65 ADC 10
 0780 85 STA 10
 0782 68 PLA
 0783 29 AND #0F
```

Morse Code Listings, cont'd.

```
<K>*=0785
/35
 0785 65 ADC 10
 0787 85 STA 10
 0789 38 SEC
 078A A2 LDX #00
 078C A9 LDA #94
 078E 85 STA 08
 0790 A9 LDA #04
 0792 85 STA 09
 0794 A5 LDA 08
 0796 E5 SBC 10
 0798 85 STA 08
 079A A5 LDA 09
 079C E9 SBC #00
 079E 85 STA 09
 07A0 E8 INX
 07A1 B0 BCS 0794
 07A3 86 STX 07
 07A5 60 RTS
 07A6 EA NOP
 07A7 EA NOP
 07A8 EA NOP
 07A9 EA NOP
 07AA EA NOP
 07AB A9 LDA #00
 07AD 8D STA A008
 07B0 A9 LDA #04
 07B2 8D STA A009
 07B5 60 RTS
 07B6 98 TYA
 07B7 09 ORA #80
 07B9 A8 TAY
 07BA B9 LDA 0400,Y
 07BD 20 JSR 0672
 07C0 20 JSR 0660
 07C3 60 RTS
```

```
 0839 06 ASL 14
 083B 26 ROL 15
 083D 06 ASL 16
 083F 26 ROL 17
 0841 06 ASL 16
 0843 26 ROL 17
 0845 0A ASL  A
 0846 46 LSR 1D
 0848 66 ROR 1C
 084A A9 LDA #00
 084C 85 STA 1A
 084E 85 STA 1B
 0850 A9 LDA #02
 0852 8D STA A00E
 0855 A0 LDY #01
 0857 AD LDA A001
 085A 58 CLI
 085B 4C JMP 085B
 085E 20 JSR 07AB
 0861 E6 INC 1A
 0863 D0 BNE 0867
 0865 E6 INC 1B
 0867 A5 LDA 1D
 0869 C5 CMP 1B
 086B 90 BCC 087F
 086D D0 BNE 0875
 086F A5 LDA 1A
 0871 C5 CMP 1C
 0873 B0 BCS 087F
 0875 A9 LDA #20
 0877 2D AND A00D
 087A F0 BEQ 0875
 087C 4C JMP 085E
 087F A5 LDA 19
 0881 C5 CMP 1D
 0883 90 BCC 08AE
 0885 D0 BNE 086D
```

```
 089E C8 INY
 089F B0 BCS 08AE
 08A1 A5 LDA 19
 08A3 C5 CMP 12
 08A5 90 BCC 08FB
 08A7 A9 LDA #A0
 08A9 A2 LDX #00
 08AB 20 JSR EF7B
 08AE A9 LDA #00
 08B0 85 STA 18
 08B2 85 STA 19
 08B4 A9 LDA #20
 08B6 2D AND A00D
 08B9 F0 BEQ 08B4
 08BB 20 JSR 07AB
 08BE E6 INC 1A
 08C0 D0 BNE 08C4
 08C2 E6 INC 1B
 08C4 A5 LDA 1B
 08C6 C5 CMP 15
 08C8 90 BCC 08B4
 08CA D0 BNE 08D2
 08CC A5 LDA 1A
 08CE C5 CMP 14
 08D0 90 BCC 08B4
 08D2 20 JSR 07B6
 08D5 A0 LDY #01
 08D7 A9 LDA #20
 08D9 2D AND A00D
 08DC F0 BEQ 08D7
 08DE 20 JSR 07AB
 08E1 E6 INC 1A
 08E3 D0 BNE 08E7
 08E5 E6 INC 1B
 08E7 A5 LDA 1B
 08E9 C5 CMP 17
 08EB 90 BCC 08D7
```

```
<K>*=0820
/50
 0820 78 SEI
 0821 A5 LDA 07
 0823 85 STA 12
 0825 85 STA 14
 0827 85 STA 16
 0829 85 STA 1C
 082B A9 LDA #00
 082D 85 STA 13
 082F 85 STA 15
 0831 85 STA 17
 0833 85 STA 18
 0835 85 STA 19
 0837 85 STA 1D
```

```
<K>*=0887
/50
 0887 A5 LDA 18
 0889 C5 CMP 1C
 088B 90 BCC 08AE
 088D 98 TYA
 088E 0A ASL  A
 088F A8 TAY
 0890 A5 LDA 19
 0892 C5 CMP 15
 0894 90 BCC 08A1
 0896 D0 BNE 089E
 0898 A5 LDA 18
 089A C5 CMP 14
 089C 90 BCC 08A1
```

```
<K>*=08ED
/50
 08ED D0 BNE 08F5
 08EF A5 LDA 1A
 08F1 C5 CMP 16
 08F3 90 BCC 08D7
 08F5 20 JSR 07B6
 08F8 4C JMP 0820
 08FB A9 LDA #A0
 08FD A2 LDX #01
 08FF 4C JMP 08AB
 0902 EA NOP
 0903 EA NOP
 0904 AD LDA A004
 0907 AD LDA A402
```

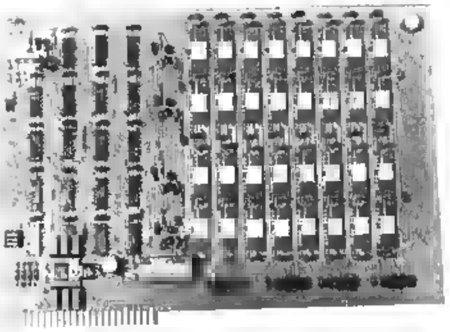
Morse Code Listings, conclusion

```
 090A C9 CMP #FF
 090C F0 BEQ 0915
 090E 0D ORA A47F
 0911 49 EOR #FF
 0913 D0 BNE 0958
 0915 A2 LDX #00
 0917 8E STX A42A
 091A 20 JSR ED05
 091D 88 DEY
 091E 30 BMI 099C
 0920 A9 LDA #8F
 0922 8D STA A400
 0925 AD LDA A402
 0928 4A LSR  A
 0929 B0 BCS 094B
 092B A2 LDX #03
 092D A9 LDA #7F
 092F 38 SEC
 0930 6A ROR  A
 0931 48 PHA
 0932 20 JSR ED0B
 0935 AD LDA A402
 0938 4A LSR  A
 0939 90 BCC 0941
 093B 68 PLA
 093C CA DEX
 093D D0 BNE 092F
 093F F0 BEQ 099C
 0941 68 PLA
 0942 AD LDA A42B
 0945 49 EOR #FF
 0947 AA TAX
 0948 EE INC A42A
 094B 20 JSR ED05
 094E 88 DEY
 094F D0 BNE 095A
 0951 AD LDA A42B

<K>*=0954
/50
 0954 C9 CMP #F7
 0956 B0 BCS 095C
 0958 90 BCC 099C
 095A 30 BMI 099C
 095C EA NOP
 095D EA NOP
 095E EA NOP
 095F 98 TYA
 0960 0A ASL .A
 0961 0A ASL .A
 0962 0A ASL .A
 0963 A8 TAY
 0964 AD LDA A42B
 0967 4A LSR .A
```

```
 0968 90 BCC 096D
 096A C8 INY
 096B D0 BNE 0967
 096D B9 LDA F421,Y
 0970 48 PHA
 0971 8A TXA
 0972 F0 BEQ 0998
 0974 29 AND #10
 0976 F0 BEQ 097E
 0978 68 PLA
 0979 29 AND #3F
 097B 4C JMP 0999
 097E 68 PLA
 097F 48 PHA
 0980 29 AND #40
 0982 D0 BNE 0998
 0984 68 PLA
 0985 48 PHA
 0986 29 AND #0F
 0988 F0 BEQ 0998
 098A C9 CMP #0C
 098C B0 BCS 0993
 098E 68 PLA
 098F 29 AND #EF
 0991 D0 BNE 0999
 0993 68 PLA
 0994 09 ORA #10
 0996 D0 BNE 0999
 0998 68 PLA
 0999 20 JSR 0600
 099C 68 PLA
 099D A8 TAY
 099E 68 PLA
 099F AA TAX
 09A0 A9 LDA #00
 09A2 8D STA A42A

<K>*=09A5
/50
 09A5 68 PLA
 09A6 40 RTI
 09A7 C9 CMP #0C
 09A9 D0 BNE 09AF
 09AB 58 CLI
 09AC 4C JMP 09AC
 09AF C9 CMP #0D
 09B1 D0 BNE 09BD
 09B3 A9 LDA #02
 09B5 8D STA A00E
 09B8 4C JMP 0B0F
 09BB C9 CMP #10
 09BD D0 BNE 09C7
 09BF AD LDA A411
 09C2 45 EOR #80
 09C4 8D STA A411
 09C7 60 RTS
```

Classified Ads

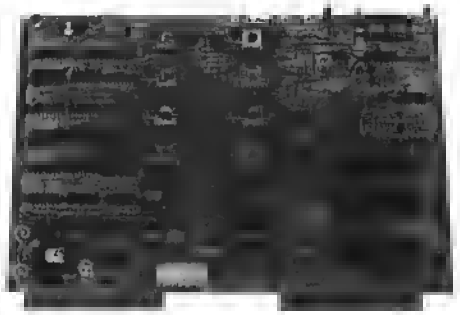
16K MEMORY



K-1016

- ADDRESSED AS CONTIGUOUS 16K STARTING AT ANY 8K BOUNDARY
- LOW POWER — 1.6 WATTS TOTAL
- K-1016A — $340 6 MONTH WARRANTY

SYSTEM EXPANSION



K-1012

- 12 PROM SOCKETS — 2708/TMS 2716, USES THE POWER OF ONLY 1 PROM.
- 32 BIDIRECTIONAL I/O LINES
- FULL RS-232 ASYNC SERIAL COMMUNICATIONS, 75-4800 BAUD
- PROM PROGRAMMER
- K-1012A — $295

EXPANSION
FOR YOUR 6502 COMPUTER

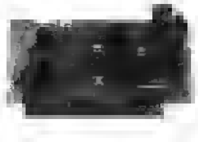
HIGH RESOLUTION GRAPHICS

- 320 x 200 BIT MAPPED GRAPHICS
- 8K RAM AVAILABLE FOR USE
- EACH POINT INDIVIDUALLY ADDRESSABLE
- K-1008A — $240, PET — $243 (PLUS PET INTERFACE)

MULTI-HARMONIC 4 VOICE MUSIC

K-1002-2

MODEL FOR ALL PETS



K-1002

MODEL FOR KIM, AIM, SYM

- FORIER SYNTHESIZED WAVEFORMS — UP TO 16 HARMONICS
- 4 VOICES PLAY SIMULTANEOUSLY
- QUALITY D/A CONVERTER, 6 POLE FILTER AND AMPLIFIER
- HARDWARE — $40.50, SOFTWARE — $20

ALL MTU PRODUCTS ARE SUPPLIED WITH FULL DOCUMENTATION CLASSED AS "BEST IN THE INDUSTRY". MANUALS MAY BE PURCHASED SEPARATELY.

MTU

Micro Technology Unlimited
P.O. Box 4596, 841 Galaxy Way
Manchester, N.H. 03108
603-627-1464

Call Or Write For Our Full Line Catalog

Skyles Electric Works

PET 2001-8 PET owners:

Do You Want Your PET To Be a Word Processor, Too?

Well, it can be...with Skyles Electric Works' new Word Processing PCB designed especially for the 2001-8. You'll need Commodore's new Disk Drive, of course. And you'll need an additional 8K of RAM memory. (We recommend the Skyles Memory Expansion System, of course.)

Skyles then supplies the interfacing PCB on which you can put the Commodore Word Processor chip. Or, even better, you can buy from Skyles the PCB complete with Commodore's Word Processor in place.

But wait: you can add the Toolkit at the same time; the Toolkit with the so important ten commands. Here's the lineup:

The PCB, to accept the Word Processor only	**$ 30.00***
The PCB, to accept both Word Processor and Toolkit	**40.00***
The PCB, to accept Word Processor; Toolkit tested and installed	**90.00***
The PCB with the Word Processor tested and installed	**140.00***
The PCB with both the Word Processor and Toolkit tested and installed.	**190.00***

From S.E.W. only: custom designed for your PET 2001-8 to interface with most memory expansion systems. Or, even better, with the 8KB Skyles Memory Expansion System.

PET LOVERS SPECIAL:
S.E.W. MEMORY EXPANSION SYSTEMS

8KB Memory Expansion System **$225**
16KB Memory Expansion System **$425**
24KB Memory Expansion System **$625**

SPECIAL PRICE WITH PURCHASE OF ANY WORD PROCESSOR OPTION ABOVE

This offer Expires February 14, 1980

2001-8 owners, you can now use your PET for word processing. Skyles Electric Works didn't forget you...

**California residents: please add 6% or 6.5% sales tax as required*

VISA, MASTERCHARGE ORDERS CALL (800) 538-3083 (except California residents)
CALIFORNIA ORDERS PLEASE CALL (408) 257-9140

Skyles Electric Works

10301 Stonydale Drive
Cupertino, California 95014
(408) 735-7891

MICRO Club Forum

MICRO is interested in having a monthly feature on 6502-related clubs. We would like to publish the names, locations and activities of groups that could be of interest to our readers. We attempted to start this feature in the past; but because of technical and publication problems, it fell by the wayside. We are now ready to get it underway.

If you are a member of such a club, have your representative register your group with us. A form for this purpose is included on our tear-out sheet. In return, we will send a free one-year subscription to MICRO for your club's library.

We would like this feature to be as helpful to our readers as possible. We welcome any information that will be of interest to other clubs; i.e., what clubs do, how they get started, what they publish, meeting format, their purpose, etc.

We are publishing as complete a list as we presently have of interested clubs. We will update it periodically, much like our bibliography section. Start increasing your membership and give your group new exposure by telling others about yourselves.

Apple Group - New Jersey
Meets the 4th Friday of every month, 7:00 p.m., at:
Union County Technical Institute
1776 Raritan Road
Scotch Plains, N.J.
Contact:Apple Group-N.J.
c/o Steve Toth
1411 Greenwood Drive
Piscataway, N.J. 08854
Tel: (201) 968-7498

The NYC User Group
The Drysdale Security
55Water Street
New York, NY 10004
Contact: Pres. Neil Shapiro
home: (516)-579-4295 (after 6 p.m.) or office: (212)269-4808

PACS PET User Group
Meets the third Saturday (ll:00 a.m.) every month in the:
Science Building
LaSalle College
20th and Onley Avenue
Philadelphia, PA 19191

Washington Apple Pi
Meets the fourth Saturday (9:30 a.m.) every month at:
George Washington University
Rm. 206, Tompkins Hall
23rd and H Streets N.W.
Washington, DC
Write: Washington Apple Pi
P.O. Box 34511
Washington, DC 20034
or call: Sandy Greenfarb, (301)674-5982
Publishes a monthly newsletter.

South Carolina Apple
Meets second Tuesdays (7:30 p.m.) at:
Byte Shop
1920 Blossom Street
Columbia, SC 29205
Write: P.O. Box 70278
Charleston Heights, SC 29405

WAKE —
Washington Area Kim Enthusiasts
Meets the third Wednesday (7:30 p.m.) of every month at:
McGraw-Hill Continuing Education Centerin Washington, D.C..
Contact: WAKE, c/o Ted Beach
5112 Williamsburg Boulevard
Arlington, VA 22207
or phone (703)538-2303

Miami Apple Users Group (M.A.U.G.)
Contact: David Hall, Secretary
2300 N.W. 135th Street
Miami, FL 33167

Sun Coast Apple Tree (SCAT)
Meets first and third Thursdays (7:00 p.m.) at:
The Computer Store
21 Clearwater Mall
Clearwater, FL 33516

COACH —
Central Ohio Apple Computer Hobbyists
Meets the third Saturday (l:00 to 5:00 p.m.)of every month
Contact: Tom Mimlitch
1547 Cunard Road
Columbus, Ohio 43227
Phone (614) 237-3380

APPLE Dayton
Meets the second Wednesday of odd numbered months and the second Thursday of even numbered months (7:30 p.m.) at:
Computer Solutions
Contact: Robert W. Rennard
2281 Cobble Stone Court
Dayton, OH 45431
Phone: (513) 426-3579

Madison Pet Users Club
Meets monthly at:
Washington Square Building
1400 East Washington Avenue
Madison, WI 53913
Contact: Ben A. Stewart
501 Willow
West Baraboo, WI 53913

Micro and Personal Computer Club of St. Louis
Meets monthly at:
Futureworld, Inc.
12304 Manchester Road
St. Louis, MO 63131
Contact: Mr. KunihiroTanaka
314) 645-4431

Tulsa Computer Society
Meets the last Tuesday (7:30 p.m.) of every month, at:
Tulsa Vo-Tech School, seminar center,
3420 S. Memorial Drive
Tulsa, OK

The Apple Corps
Meets the second Saturday (2-5 p.m.) of each month at:
Greenhill School
14255 Midway Road
Dallas, TX

Apple User Group
Meets the second Tuesday of each month at:
High Technology of Tulsa Computer Store
2601D S. Memorial Drive, Tulsa

For information on both ot the above gioups, wiite:
The Tulsa Computei Society
P.O. Box 1133
Tulsa, OK 74101

Appleseed
Meets monthly at:
The Computer Shop
6812 San Pedro
San Antonio, TX 78216
(512) 828-0553

The Austin Apple Corps
Meets first and third Tuesdays (7:00 p.m.) ot every month.

Contact: Mike Palmore, (512) 442-4871/447-0332; Kris Cobb (512)837-7228/443-7711; or Lenny Fein (512)441-3220/471-1154.

The L.A. Apple Useis Group
Meets the first Friday (7:30 p.m.) ot every month at:
Allstate Savings Community Room
8800 S. Sepulveda Boulevard
Los Angeles, CA.

Contact: Philip A. Wasson
9513 Hindry Place
Los Angeles, CA 90045

The San Fernando Valley 6502 Users Club
Meets the second Tuesday (8:00 p.m.) of every month at:
Computer Components Inc. ot Burbank
3808 West Verdogo Avenue
Burbank, CA 91505

Contact: Larry Goga
3816 Albright Avenue
Los Angeles, CA 90066

Publishes a monthly newsletter.

Honolulu Apple Users Society
Meets the first Monday of each month at:
Computerland Store in Honolulu.

Contact: Bill Mark
98-1451-A Kaahumanu Street
Aiea, Hawaii 96701
Phone: (808)488-2026

Northwest Pet User's Group
Contact: John F. Jones
2134 NE 45th Avenue
Portland, OR 97213
Phone: (503)281-4908

Northwest Pet Useis Group
Meets monthly at:
Seawel Marketing
315 B N.W. 85
Seattle, WA

Contact: Jeffrey Dukes
15346 SE 307
Kent, WA 98031
Phone: (206)631-1973

A.P.P.L.E.
Meets the third Tuesday of every month.

Contact: Ralph Thiers
8710 Salty Drive, NW
Olympia, WA 98502

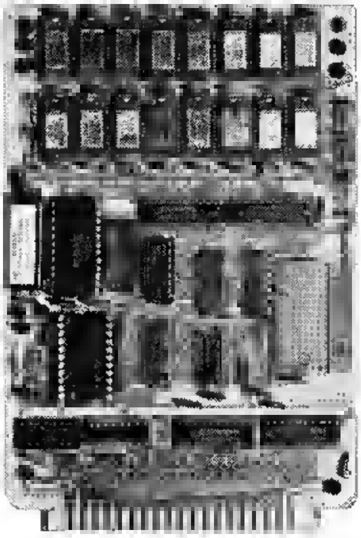
Apples British Columbia Computer Society
Meets the first Wednesday of every month.

Contact: Gary B. Little
101-2044 West Third Avenue
Vancouver, British Columbia,
Canada V6J 1L5
Phone: (604)731-7886

The North London Hobby Computer Club
Different gioups within the club meet many times weekly, in addition to a club-wide monthly meeting.

Contact: Stephanie Bromley
The Polytechnic ot North London
Holloway, London N7 8DB
Phone: 01-607-2789

DeVlaamse Minicomputerclub V2W
Lambrechtshoekenlaan 171b6
2060 Merksem, Belgium



The Great Superboard Speed-Up and Other RAMblings

I do not know if Bufferin is twice as fast as Aspirin, but here is all you need to make your OSI Model 600 board run twice as fast as it normally does.

Jack Robert Swindell
P.O. Box 8193
Canton, OH 44711

The OSI Superboard 11, Challenger 1P is a great machine — tast so you can really get the job done. Not bad considering that it is running at under 1MHZ. Wouldn't it be even nicer running at 2? Don't start jumping up and down and barking yet, we have a few hurdles to jump tirst. They are not really tall ones, but you had better know where they are at instead ot stumbling into them.

The model 600 board was designed to run the 6502 at about 983KHZ or almost 1MHZ. This meant that they could keep the cost down by having highly efficient software resident in ROM's (firmware) do the magic of making process time short instead ot sloppy sottware with a taster clock rate to help make up tor it. The cost saving is in the RAM...it only seems to be good for 1MHZ or thereabouts. Apparently the same Basic in ROM is used in several OSI computers with the I/O handling controlled by a monitor/support ROM unique to each model (or series). It this really is the case (does anyone know tor sure?) then the Basic in ROM must be able to operate at 2MHZ to prevent having to stock multiple grades ot ROM (which is a rather expensive proposition) for the ditterent speeds of CPU's.

The other thing that makes me think that there is only one grade ot Basic in ROM is that there are no sutfix marks on the ROM's to indicate that they might have been sorted tor speed. It is possible that the monitor/support ROM was only specified to guarantee operation at 1MHZ as that is the intended processor operating speed tor the 600 series board. As this ROM is probably unique to the model 600 and would not appear on the 2MHZ board, the 2MHZ capability may not have been specified tor this chip.

There is one other thing to consider before delving into the hardware aspect ot this project. Do you have any optional boards tied into your 600 board? Especially memory...the original tactory-installed RAM on my card was not able to make 2MHZ; theretore, I most certainly wouldn't count on their expansion RAM handling double the normal recommended speed. Translated: The memory that you already have probably won't work at 2MHZ and will have to be replaced (OUCH). Perhaps you could trade with someone. Well, let's not jump the gun and start ordering parts yet, there is always that chance that *your* memory might be different than mine and will work OK...I hope so. My originals were 2114L's by SEMMI. I don't know what happens if you have a mini-tloppy tied in and then double the speed. Also assume that your warranty is shot once you modify it. You might want to wait until it expires.

The first thing to do is to decide whether or not you want to go any further than just reading this article.*Remember:* Neither the author nor MICRO guarantee the satety or operation of this moditication, nor should you expect the manutacturer or service department to honor any warranties atter you have modified your equipment. Mostly what I am saying is that if you don't understand what you are doing: *DON'T DO IT!* And...if you goof up and ruin your machine *you did it yourself.* I don't know how to say it in proper legalese, but you get the picture.

TURN OFF THE POWER FIRST!!!!

The illustration applies to my model 600 CPU, revision B. What this modification is doing is moving the tap on the clock circuit divider chain one divide by two closer to the oscillator. You're sure that you want to do this? OK...cut the line as shown in the illustration. You have just severed the clock line going to pin 37 on the 6502. Take a small piece of insulated wire and make a jumper like in the illustration. You won't have to strip off very much insulation at each end to do the job. Solder it in, again see the illustration, taking care not to short any of those eentsy conductors nearby. Now the CPU will have twice the clock speed as before. Now to see how it turned out.

I hope your memory makes it as is...we'll soon see. Connect the video monitor cable and turn on the monitor. Do not connect any oft-card peripherals ot any sort yet. Now apply power to the CPU and press BREAK. Does the screen show any characters other than D/C/W/M? It so, jump to the next paragraph. Press C and tinish off the usual initialization routine. If there are any incorrect characters, jump to the next paragraph. Try to run a tew simple two or three line programs and solve some easy problems in the command mode. If anything didn't work satistactorily, jump to the next paragraph. Congratulations, you are now the owner of a super-Superboard. Keep an eye open just in case any problems might develop until you feel sure that all is OK. Branch to the next sub-heading.

It you are reading this paragraph then you have a minor problem to solve. Most probably your RAM is a bit too slow. Try to borrow four 2114 RAM's known to be good at 2 or more MHZ. Pull out all ten (or eighteen) RAM's on your CPU card (note polarity), both program and video memory. Look in the back of your User's manual for the locations ot U31, U39, U40 and U45. Plug in the taster 2114's here making sure that you get them in the same way that the others came out. Try to run through the initialization tests ot the previous prargraphs. It should say that it has 255 bytes free. If this doesn't work, you can either try one more set of different RAM's in the hope that one ot them still wasn't fast enough. No go? I'm sorry...probably one ot the ROM's is a bit slow. Well, just reverse the order of steps in the moditication, restore the original memory chips (making sure to put a jumper in where you cut the line and removing your modification jumper) and you're none the worse tor wear.

COMMAND MODE STRING PRINTING

I have one small item of curiosity to throw in before I vector off into oblivion. Type (in command mode) ?"67 or 68 characters", press RETURN. It may or may not print the string and will almost always print a syntax error at some non-existent line number. Branch to next article.

HAPPY COMPUTING!

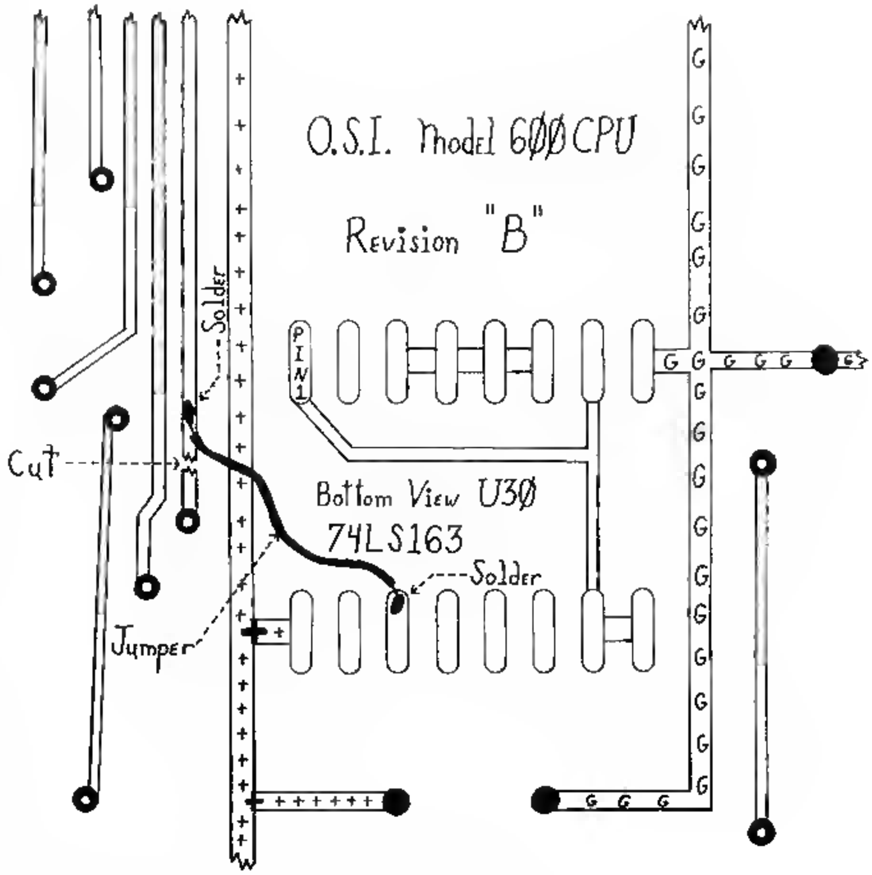


Figure 1

SUPER-TEXT™

STANDARD FEATURES

- single key cursor control
- automatic word overflow
- character, word and line insertion
- forward and backward scrolling
- automatic on screen tabbing
- single key for entering "the"
- auto paragraph indentation
- character, word and line deletion
- ditto key
- multiple text windows
- block copy, save and delete
- advanced file handling
- global (multi-file) search and replace
- on screen math and column totals
- column decimal alignment
- chapter relative page numbering
- complete printer tab control
- line centering
- superscripting and subscripting
- two color printing
- underscoring and boldface
- user defined special functions

The Professional Word Processor

for the Apple II
and the Apple II plus

MUSE™
THE LEADER IN DUALITY SOFTWARE

FAST EDITING

Super-Text was designed by a professional writer for simple, efficient operation. A full floating cursor and multiple text screens facilitate editing one section of text while referencing another. Super-Text's advanced features actually make it easier to operate, allowing you to concentrate on writing rather than remembering complicated key sequences.

FLDATING PDINT CALCULATDR

A built in 15 digit calculator performs on-screen calculations, column totals and verifies numeric data in statistical documents.

EXCLUSIVE AUTDLINK

Easily link an unlimited numher of on-line files on one disk or from disk to disk. Autolink allows you to search or print all on-line files with a single command. Typical files of items that can be stored in this way include personnel files, prospect files, maintenance records, training records and medical histories.

ADVANCED FILE HANDLING

Single key file manipulation and complete block operations allow the user to quickly piece together stored paragraphs and phrases. Text files are listed in a directory with a corresponding index for fast and accurate text retrieval.

PRINTER CDNTRDLS

Super-Text is compatible with any printer that interfaces with an Apple. Print single or multiple copies of your text files or link files and they will be automatically printed in the specified order. User defined control characters can activate most special printer functions.

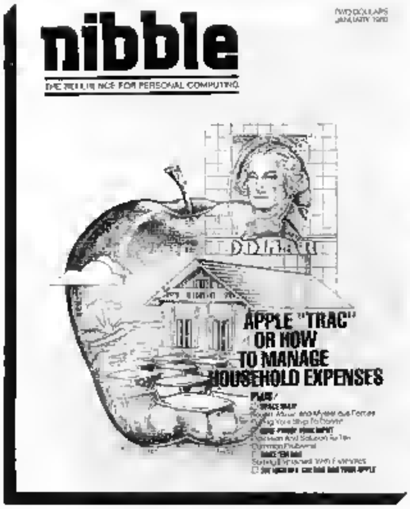
MDDULAR DESIGN

This is a modularly designed system with the flexibility for meeting your future word processing needs. The first add-on module will be a form letter generator for matching mailing lists with Super-Text form letters. The form letter module will be available in the first quarter of 1980.

SUPER-TEXT, requires 48K ($99.95)
Available TODAY at Computer Stores
nationwide. Dealer inquiries welcome. For more
information write:

MUSE SOFTWARE 330 N. Charles Street Baltimore, MD 21201 (301) 659-7212

INTRODUCING . . . NIBBLE
THE REFERENCE
FOR APPLE COMPUTING



NIBBLE IS:
A SOFTWARE GUIDE for high quality Applications Programs for your Home and Business.

NIBBLE IS:
A REFERENCE GUIDE to new Programming Methods.

NIBBLE IS:
A BUYERS GUIDE for making purchase decisions on new products.

NIBBLE IS:
A CONSTRUCTION PROJECT COOKBOOK for adding function and value to the system you already own.

NIBBLE IS:
A COMMUNICATIONS CLEARING HOUSE for users, vendors, and associations.

Each issue of NIBBLE features at least one significant new application program of commercial quality. The programs in NIBBLE are surrounded with articles which show how to USE the programming methods in your OWN programs.

Examples of upcoming articles:
□ Modeling and Forecasting Your Business □ Build a Two-Tape Controller for $12
□ Arcade Shooting Gallery — Save Your Quarters! □ Data Base Management System I, II, III

And many many more! NIBBLE will literally "Nibble Away" at the mysteries of your system to help you USE IT MORE. In 1980, the principal featured system is the Apple II.

Try a NIBBLE

nibble
BOX 325 Lincoln, Mass. 01773

I'll try NIBBLE !
Enclosed Is my $15 for 8 issues.
□ **check** □ **money order**

Name ______________________________

Address ______________________________

City ______________________________

State ________________ Zip ________

KIM-1 Tape Recorder Controller

Some techniques for using a 6502 micro for controlling switches are presented. The particular application is for a KIM to control a tape deck, but the concepts are quite broad in scope.

Michael Urban
General Electric
SPD Box 43
Auburn, NY 13021

OBJECTIVE

The Kim-1 microcomputer is to be used to control the four functions (play, rewind, wind and stop) of a Tandbert 9000X open-reel tape deck by way of the remote control socket at the back of the deck. This control will enable the user to program the computer to automatically locate and play a sequence of songs previously selected.

METHOD

The heart of the operating program is the tape counter displayed on the address LED's which simulate the mechanical tape counter on the deck itself. The actual program increments or decrements this counter, compares the desired location to the present counter, and then directs the tape deck on the result ot that comparison. A description of each of the blocks of the program tlow chart follows:

Initialization-

Here the counter, data register, and x and y registers are cleared. The data direction register is set to FF tor an output condition. the x-register is loaded with the tirst song selection at location 0000 plus the y-register. The contents ot both registers are then saved, using a STORE subroutine.

Compare-

The high order byte of the counter (OOFB) is compared with the contents ot location 0050 plus the x-register. This location is reserved for the high order bytes of any song starting location. If the result is either positive or negative, the program branches to wind or rewind respectively. If the result is zero, the low order byte must be compared. Because ot differing branch instructions, there are separate wind compares and rewind compares. Each of these takes the low order bytes ot the counter (OOFA) and compares it to the contents of location 0060 plus the x-register. The program then goes to either wind, rewind or play, depending on the results.

Wind-

A 08 is placed in the data register fo put the tape deck in the wind mode. The tape counter is incremented by adding 01 to 00FA. A delay loop is set up with the interval timer and the counter displayed using the SCANDS subroutine. Jump to cmp.

Rewind-

A 01 is placed in the data register to put the tape deck in the rewind mode. The tape counter is decremented by subtracting 01 from 00FA. A delay loop is again set up with the interval timer and the counter displayed using the SCANDS subroutine. Jump back to Compare.

Stop/Wait-

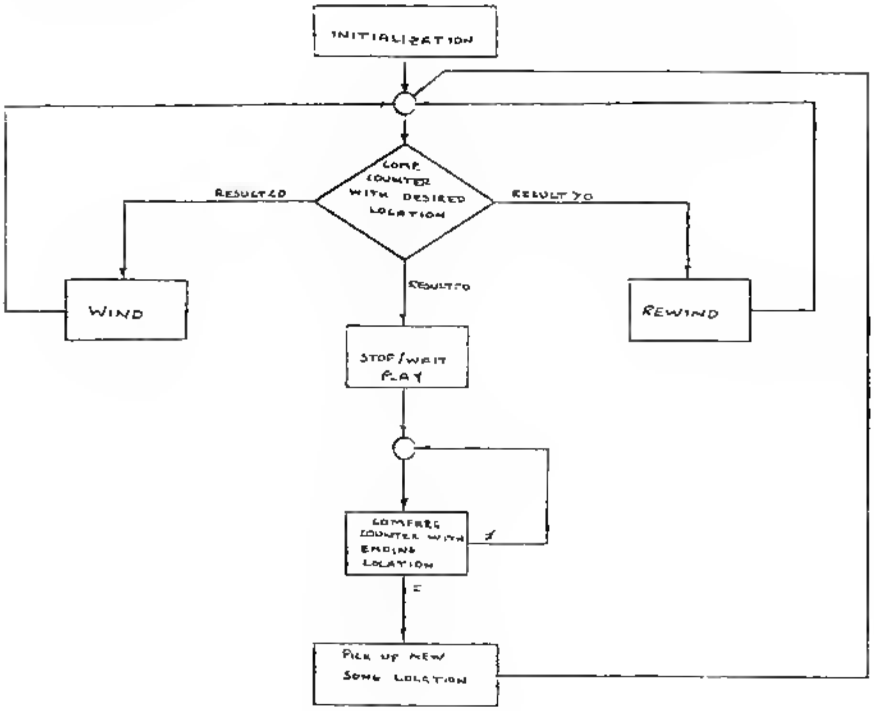
A 04 is stored in the data register to stop the tape deck. Another delay loop is utilized to wait for the deck to come to a halt before putting it in the play mode. The counter is displayed on the LED's.

Play-

The contents of the x-register are placed in 00F9 so that the next display will show the song selection while playing it. A 02 is placed in the data register to put the tape deck in the play mode. The counter is incremented by adding 01 to 00FA. A delay loop is set up using the interval timer. The high order byte ot the counter is now compared to the contents of location 0070 plus the x-register. This is the location of the ending location ot the selected song, high order byte. If the high order bytes are not equal, the program branches back to Play. If the high order bytes *are* equal, the low order bytes must be compared. The contents of the low order byte of the counter (OOFA) are now compared to the contents of the address 0080 plus the x-register which is the address of the ending location, low order byte, ot the selected song. It the low order byte comparison results in a zero, the end of a song has been reached. The program sits In a delay loop waiting tor the deck to catch up. The y-register is then incremented so that the next song selection can be made. Jump back to Begin.

The Interface

Through experimentation with the remote control socket, it was tound that a short between any of the function pins and ground would cause the deck to operate In that mode. A current of 2mA was measured with a short circuit to ground. Later, it was found that a resistor to ground also worked. With 2K between the function pin and ground, a lower current of 1mA was obtained. This was ideal for our purposes. Relays were considered as the interface element but rejected because of cost and layout considerations.


INITIALIZATION
COMP. COUNTER WITH DESIRED LOCATION
RESULT<0
RESULT>0
RESULT=0
WIND
REWIND
STOP/WAIT PLAY
COMPARE COUNTER WITH ENDING LOCATION
≠
=
PICK UP NEW SONG LOCATION

REEL-TO-REEL INTERFACE

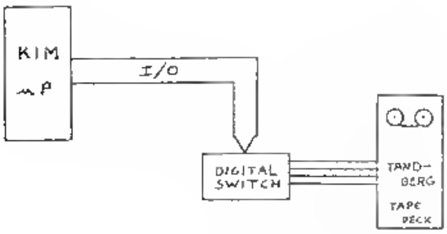


FIGURE 1

The 4016 CMOS analog/digital switch was decided upon. It is an integrated circuit containing tour independent switches ot the configuration in figure 3. An overall view of the basic interface Is pictured in figure 1. The actual wiring dlagram is seen in tigure 2. A 5-volt signal coming from any of the outputs PAO-PA3 will cause a switch ctosure in the following order:

PA0-Rewind (01)
PA1-Play (02)
PA2-Stop (04)
PA3-Wind (08)

The numbers in parenthesis indicate the number that must be in the data register tor that particular function to be performed. The resistors in tigure 2 are for current limiting through the switch.

SUMMARY

For the most part, the project was a success. The only problem encountered was that of trying to synchronize the simulated tape counter speeds to those of the mechanical one on the tape deck. To better explain this, tigure 4 is helpful. As can be seen in figure 4a, the KIM's tape counter is a very linear device unlike that of the deck's very non-linear counter in figure 4b. In the wind or rewind modes, the two ceuld never be matched because ot this non-linearity. Theretore, it was decided upon to only demonstrate the program's ability to control the tape deck and locate selections on the computer tape counter. This the program did well.

The ultimate way to circumvent this problem would be to actually couple the computer to the tape deck through an optical or magnetic pick-up on one of the tape reels. In this way, the KIM would always know preclsely where the tape was located. If, for some reason, this was not possible, a linear approximation could be programmed into the computer to simulate the acceleration curve of the mechanical tape counter. This would consist of three or four loops ot dittering speeds cascaded together to form a curve like that ot tigure 4c.

In recent years, commercial manufacturers have been incorporating a similar program-locating teature into cassette decks. The most notable is the Sharp RT-3388A which has its own dedicated microprocessor which will locate a particular section of the tape requested and plays trom there on; it does not have the ability of playing any sequence of songs asked tor by the user. In this respect, our program is superior.

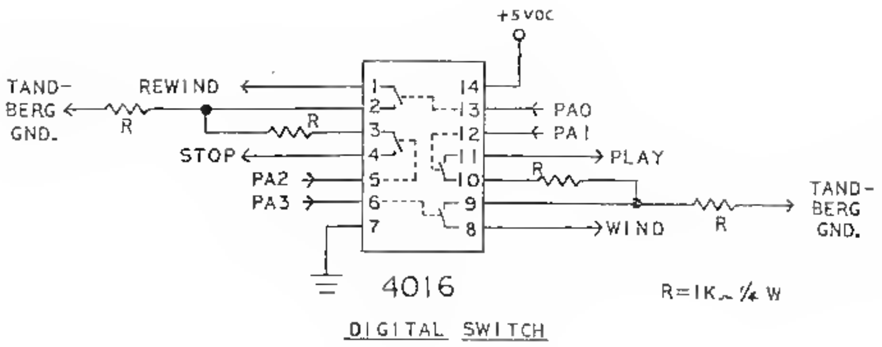


FIGURE 2

INTERNAL SCHEMATIC

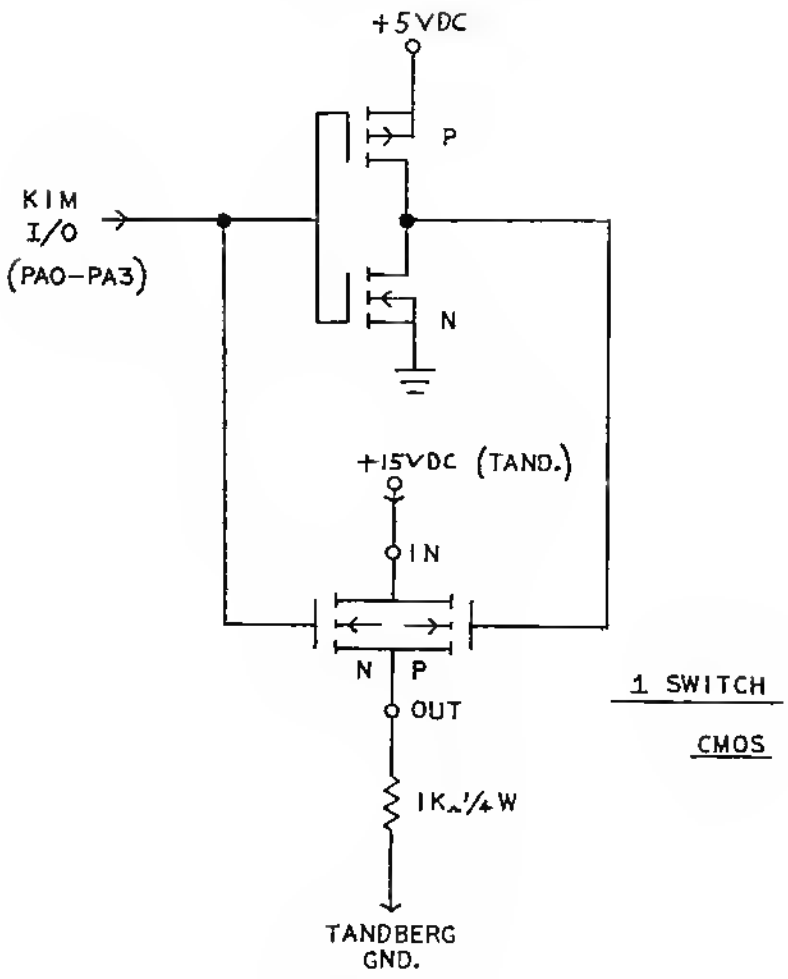
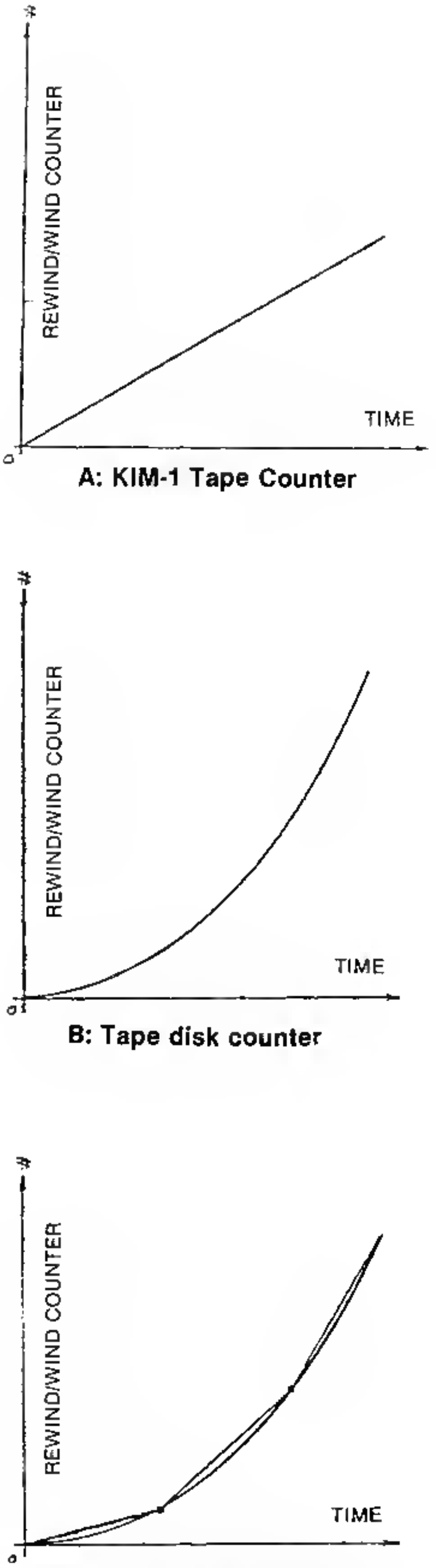


FIGURE 3



A: KIM-1 Tape Counter

B: Tape disk counter

C: Linear approximation

Figure 4

bytes must be compared. The contents of the low order byte of the counter (OOFA) are now compared to the contents of the address 0080 plus the x-register which is the address of the ending location, low order byte, of the selected song. If the low order byte comparison results in a zero, the end of a song has been reached. The program sits in a delay loop waiting for the deck to catch up. The y-register is then incremented so that the next song selection can be made. Jump back to Begin.

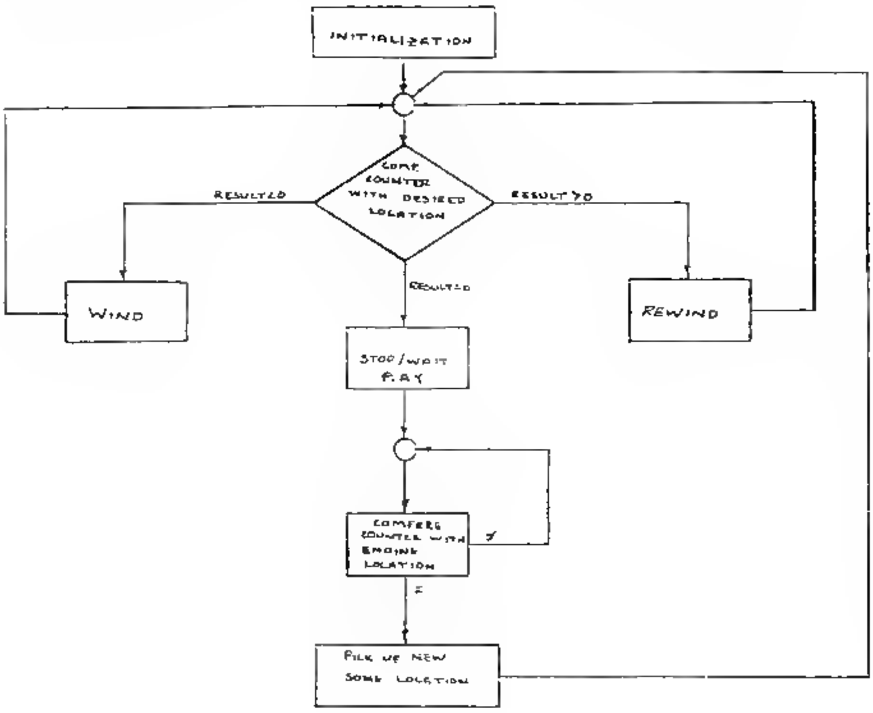
The Interface

Through experimentation with the remote control socket, it was found that a short between any of the function pins and ground would cause the deck to operate in that mode. A current of 2mA was measured with a short circuit to ground. Later, it was found that a resistor to ground also worked. With 2K between the tunction pin and ground, a lower current of 1mA was obtained. This was ideal for our purposes. Relays were considered as the intertace element but rejected because of cost and layout considerations.

The 4016 CMOS analog/digital switch was decided upon. It is an integrated circuit containing tour independent switches ot the configuration in tigure 3. An overall view ot the basic Intertace Is pictured in figure 1. The actual wiring diagram is seen in figure 2. A 5-volt signal coming from any ot the outputs PAO-PA3 will cause a switch closure in the tollowing order:

PA0-Rewind (01)
PA1-Play (02)
PA2-Stop (04)
PA3-Wind (08)

The numbers in parenthesis Indicate the number that must be in the data register tor that particular function to be performed. The resistors in figure 2 are for current limiting through the switch.


INITIALIZATION
COMP. COUNTER WITH DESIRED LOCATION
RESULT<0
RESULT>0
RESULT=0
WIND
REWIND
STOP/WAIT PLAY
COMPARE COUNTER WITH ENDING LOCATION
≠
=
PICK UP NEW SONG LOCATION

REEL-TO-REEL INTERFACE

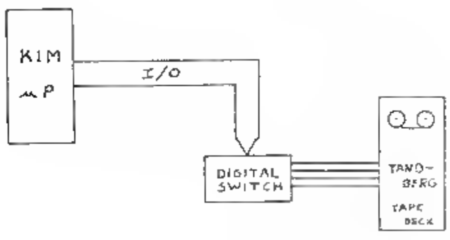


FIGURE 1

SUMMARY

For the most part, the project was a success. The only problem encountered was that ot trying to synchronize the simulated tape counter speeds to those ot the mechanical one on the tape deck. To better explain this, figure 4 is helptul. As can be seen in tigure 4a, the KIM's tape counter is a very linear device unlike that ot the deck's very non-linear counter in figure 4b. In the wind or rewind modes, the two could never be matched because of this non-linearity. Theretore, it was decided upon to only demonstrate the program's ability to control the tape deck and locate selections on the computer tape counter. This the program did well.

The ultimate way to circumvent this problem would be to actually couple the computer to the tape deck through an optical or magnetic pick-up on one ot the tape reels. In this way, the KIM would always know precisely where the tape was located. If, for some reason, this was not possible, a linear approximation could be programmed into the computer to simulate the acceleration curve ot the mechanical tape counter. This would consist of three or four loops ot diftering speeds cascaded together to form a curve like that of figure 4c.

In recent years, commercial manutacturers have been incorporating a similar program-locating feature into cassette decks. The most notable is the Sharp RT-3388A which has its own dedicated microprocessor which will locate a particular section of the tape requested and plays from there on; it does not have the ability of playing any sequence of songs asked for by the user. In this respect, our program is superior.

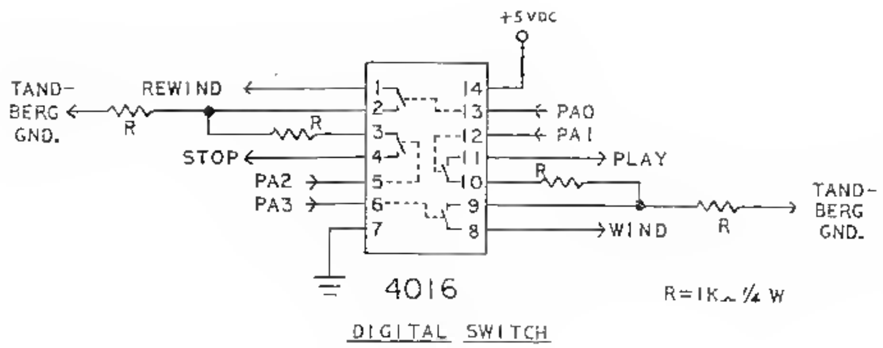


FIGURE 2

INTERNAL SCHEMATIC

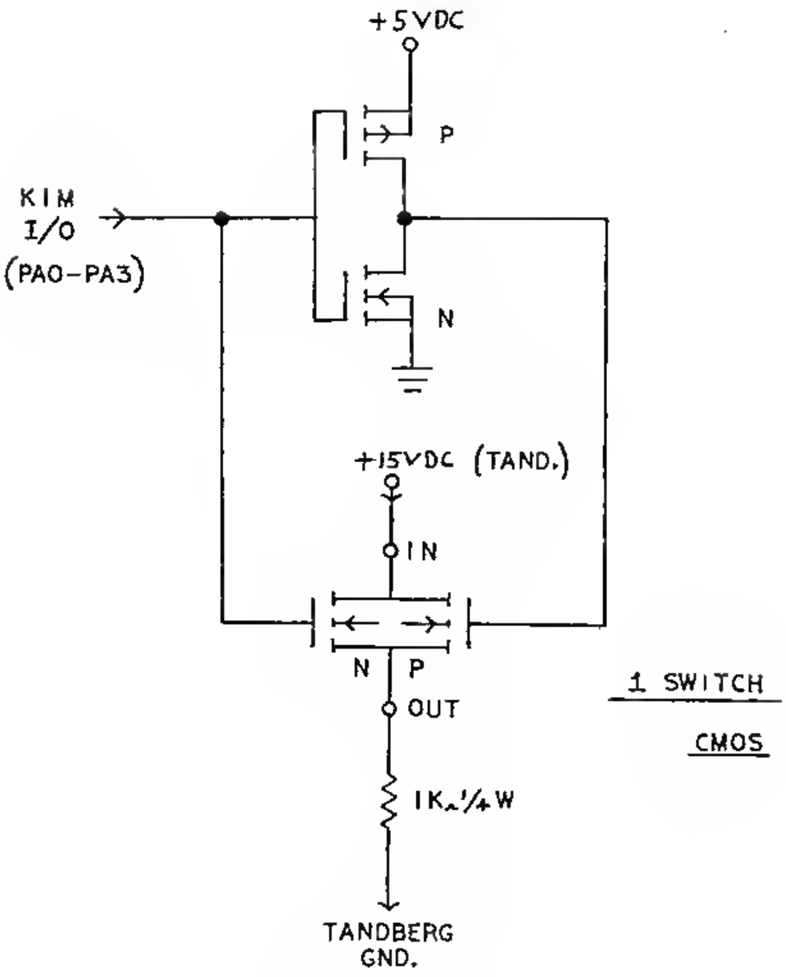
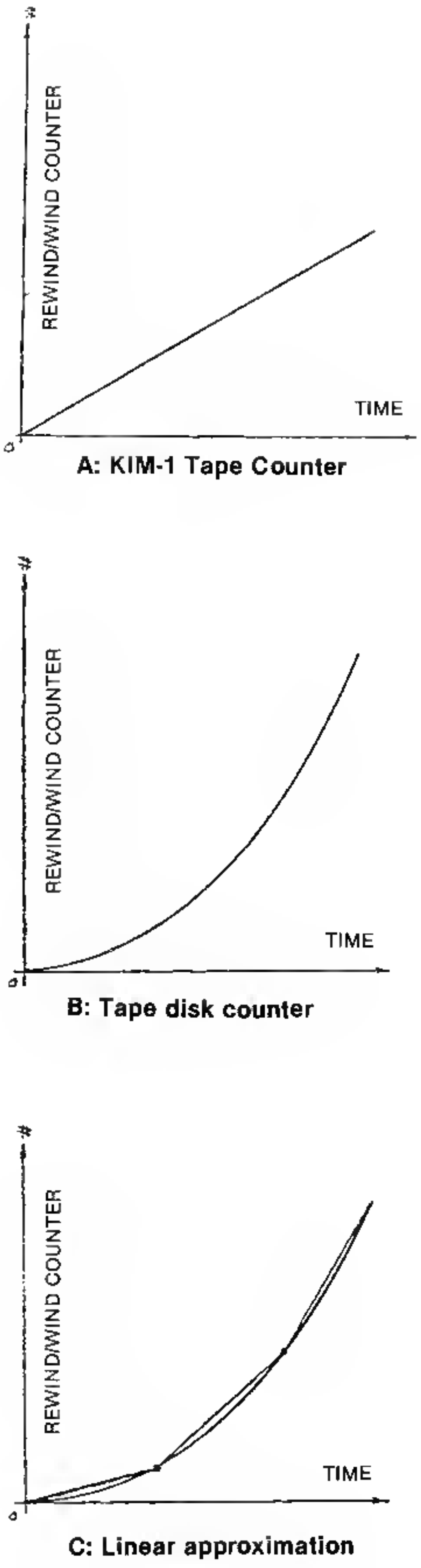


FIGURE 3



A: KIM-1 Tape Counter

B: Tape disk counter

C: Linear approximation

Figure 4

```
Address  Instruction  Label    Op Code  Operand

0210     F8                    SED
0211     18                    CLC
0212     A9 00                 LDA      #$00
0214     85 20                 STA      0020
0216     85 21                 STA      0021
0218     85 23                 STA      0023
021A     85 F9                 STA      00F9
021C     85 FA                 STA      00FA
021E     85 F8                 STA      00FB
0220     8D 00 17              STA      PAD
0223     A9 01                 LDA      #$01                INITIALIZE
0225     85 22                 STA      0022
0227     A9 FF                 LDA      #$FF
0229     8D 01 17              STA      PADD.
022C     A2 00                 LDX      #$00
022E     A0 00                 LDY      #$00
0230     B6 00        BEGIN    LDX,Y    00D0
0232     20 20 03              JSR      STORE
0235     4C 3E D2              JMP      COMPHI
023B     20 54 03     *PULL    JSR      PULL
023C     20 70 03              JSR      STORE
023E     A5 FB        COMPHI   LDA      FB
0240     D5 50                 CMP,X    0050
0242     F0 05                 BEQ      COMPLO
0244     10 10                 BPL      REWIND
0246     4C 74 02              JMP      WIND
0249     A9 01        COMPLO   LDA      #$01
024E     2D 00 17              AND      PAD
024E     F0 0C                 BEQ      WINDC
0250     A5 FA                 LDA      FA                  COMPARE
0252     D5 60                 CMP,X    0060
0254     F0 03                 BEQ      PLAY
0255     4C A5 02     REWIND   JMP      REWIND
0259     4C D2 02     PLAY     JMP      STOP
025C     A5 FF        WINDC    LDA      FA
0255     D5 60                 CMP,X    0060
0260     F0 F7                 BEQ      PLAY
0262     4C 74 02              JMP      WIND
0274     18           WIND     CLC
0275     A9 08                 LDA      #$08
0277     8D 00 17              STA      PAD
027A     A5 22                 LDA      0022
027C     65 FA                 ADC      FA
027E     85 FA                 STA      00FA
0280     A5 21                 LDA      0021
0282     65 FB                 ADC      FB
0284     85 FB                 STA      00FB
0286     A9 01                 LDA      #$D1                WIND
0288     85 25                 STA      0025
028A     A9 3E        LOOP 1   LDA      #$5F
028C     8D 07 17              STA      ÷1024I.T
028F     20 1F 1F     DISPLAY  JSR      SCANDS
0292     2C 07 17              BIT      1.T.
0295     10 F8                 BPL      DISPLAY
0297     C6 25                 DEC      0025
0299     D0 EF                 BNE      LOOP1
029B     EA EA                 NOP
029D     4C 38 02              JMP      *PULL
02A5     38           REWIND   SEC
02A6     A9 01                 LDA      #$01
02A8     8D 00 17              STA      PAD
02A8     A5 FA                 LDA      FA
02AD     E5 22                 SBC      0022
02AF     85 FA                 STA      00FA                REWIND
02B1     A5 FB                 LDA      FB
02B3     E5 23                 SBC      0023
02B5     85 FB                 STA      00FB
02B7     A9 01                 LDA      #$01
02B9     85 24                 STA      0024
```

```
Address  Instruction  Label    Op Code  Operand

02BB     A9 5F        LOOP2    LDA      #$5F
02BD     8D 07 17              STA      ÷1024I.T.
02C0     20 1F 1F     DISPLAY  JSR      SCANDS
02C3     2C 07 17              BIT      I.T.
02C6     10 F8                 BPL      DISPLAY             REWIND (cont'd)
02C8     C6 24                 DEC      0024
02CA     D0 EF                 BNE      LOOP2
02CC     38                    SEC
02CD     4C 38 02              JMP      *PULL
02D2     8A           STOP     TXA
02D3     85 F9                 STA      F9
02D5     A9 04                 LDA      #$04
02D7     8D 00 17              STA      PAD
02DA     A9 0A                 LDA      #$0A
02DC     85 26                 STA      0026
02DE     A9 FF        LOOP3    LDA      #$FF
02E0     8D 07 17              STA      ÷1024I.T.
02E3     20 1F 1F     DISPLAY  JSR      SCANDS              STOP/WAIT
02E6     2C 07 17              BIT      I.T.
02E9     10 F8                 BPL      DISPLAY
02EB     C6 26                 DEC      0026
02ED     D0 EF                 BNE      LOOP3
02EF     A9 02                 LDA      #$02
02F1     8D 00 17              STA      PAD
02F4     18                    CLC
02F5     A5 22        PLAY     LDA      0022
02F7     65 FA                 ADC      FA
02F9     85 FA                 STA      00FA
02FB     A5 21                 LDA      0021
02FD     65 FB                 ADC      FS
02FF     85 FB                 STA      00FB
0301     A9 04                 LDA      #$01
0303     85 27                 STA      0027
0305     A9 85        LOOP4    LDA      #$FF
0307     80 07 17              STA      ÷1024I.T.
030A     20 1F 1F     DISPLAY  JSR      SCANDS
030D     2C 07 17              BIT      I.T.
0310     10 F8                 BPL      DISPLAY
0312     C6 27                 DEC      0027
0314     D0 EF                 BNE      LOOP4
0316     20 54 03              JSR      PULL
0319     20 70 03              JSR      STORE
031C     A5 FB                 LDA      00FB
031E     D5 70                 CMP,X    0070                PLAY
0320     D0 D3                 BNE      PLAY
0322     A5 FF                 LDA      00FA
0324     D5 80                 CMP,X    0080
0326     D0 CD                 BNE      PLAY
0328     A9 04                 LDA      #$04
032A     8D 00 17              STA      PAD
032D     4C 30 02              JSR      PULL
0330     EA                    NOP
0331     A9 0A                 LDA      #$0A
0333     B5 26                 STA      0025
D335     A9 AF        LOOP5    LDA      #$AF
0337     8D 07 17              STA      ÷10 -1I.1
033A     20 1F 1F     DISPLAY  JSR      SCANDS
0330     2C 07 17              BIT      1.T.
0340     10 F8                 BPL      DISPLAY
0342     C6 25                 DEC      0025
0344     D0 EF                 BNE      LOOP5
0346     C8                    INY
0349     4C 30 02              JMP      BEGIN
0354     68           PULL     PLA
0355     85 30                 STA      0030
0357     68                    PLA                         SUBROUTINE
0358     85 31                 STA      0031                PULL
035A     68                    PLA
035B     A8                    TAY
```

```
Address Instruction   Label   OP Code  Operand

035C    68                    PLA
035D    AA                    TAX
035E    A5 31                 LDA      0031
0360    48                    PHA
0361    A5 30                 LDA      0030
0363    48                    PHA
0364    60                    RTS

0370    68            STORE   PLA
0371    85 30                 STA      0030
0373    68                    PLA
0374    85 31                 STA      0031
0376    8A                    TXA
0377    48                    PHA
0378    98                    TYA
0379    48                    PHA
037A    A5 31                 LDA
037C    46                    PHA
037D    A5 30                 LDA
037F    48                    PHA
0380    60                    RTS
```

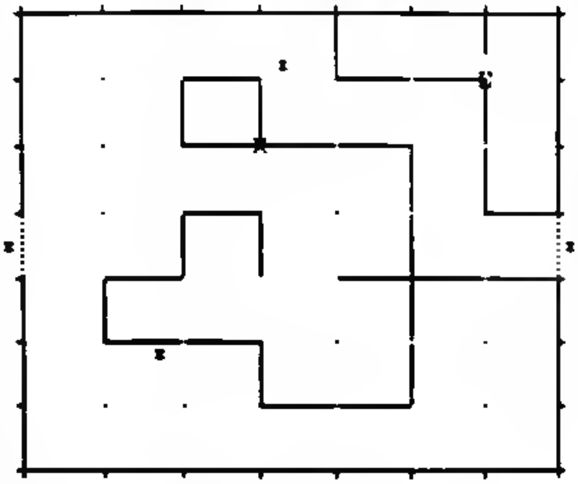
SUBROUTINE STORE

```
MEMORY ALLOCATION

Address       Label

0051          Selections 1-15
 ↓
005F          Starting Locations High

0061          Selections 1-15
 ↓
006F          Starting Locations Low

0071          Selections 1-15
 ↓
0007F         Ending Locations High

0081          Selections 1-15
 ↓
008F          Ending Locations Low

0000
 ↓            Song Sequence Numbers
000F

0210
 ↓            Operating Program
0380

0020
 ↓            Miscellaneous Locations Used
0032          Within Main Program
```

```
        Ø1ØØ
Ø1ØØ DB G

        ØØ   Ø1   Ø2   Ø3   Ø4   Ø5   Ø6   Ø7   Ø8   Ø9   ØA   ØB   ØC   ØD   ØE   ØF

Ø21Ø    F8   18   A9   ØØ   85   2Ø   85   21   85   23   85   F9   85   FA   85   FB
Ø22Ø    8D   ØØ   17   A9   Ø1   85   22   A9   FF   8D   Ø1   17   A2   ØØ   AØ   ØØ
Ø23Ø    B6   ØØ   2Ø   7Ø   Ø3   4C   3E   Ø2   2Ø   54   Ø3   2Ø   7Ø   Ø3   A5   FB
Ø24Ø    D5   5Ø   FØ   Ø5   1Ø   1Ø   4C   74   Ø2   A9   Ø1   2D   ØØ   17   FØ   ØC
Ø25Ø    A5   FA   D5   6Ø   FØ   Ø3   4C   A5   Ø2   4C   D2   Ø2   A5   FA   D5   6Ø
Ø26Ø    FØ   F7   4C   74   Ø2   A9   5F   8D   Ø7   17   2Ø   1F   1F   2C   Ø7   17
Ø27Ø    1Ø   F8   C6   25   18   A9   Ø8   8D   ØØ   17   A5   22   65   FA   85   FA
Ø28Ø    A5   21   65   FB   85   FB   A9   Ø1   85   25   A9   1E   8D   Ø7   17   2Ø
Ø29Ø    1F   1F   2C   Ø7   17   1Ø   F8   C6   25   DØ   EF   EA   EA   4C   38   Ø2
Ø2AØ    EA   EA   EA   EA   EA   38   A9   Ø1   8D   ØØ   17   A5   FA   E5   22   85
Ø2BØ    ØØ   ØØ   ØØ   ØØ   ØØ   ØØ   FB   A9   Ø1   85   24   A9   5F   8D   Ø7   17
Ø2CØ    2Ø   1F   1F   2C   Ø7   17   1Ø   F8   C6   24   DØ   EF   38   4C   38   Ø2
Ø2DØ    18   A5   8A   85   F9   A9   Ø4   8D   ØØ   17   A9   ØA   85   26   A9   FF
Ø2EØ    8D   Ø7   17   2Ø   1F   1F   2C   Ø7   17   1Ø   F8   C6   26   DØ   EF   A9
Ø2FØ    Ø2   8D   ØØ   17   18   A5   22   65   FA   85   FA   A5   21   65   FB   85
Ø3ØØ    FB   A9   Ø4   85   27   A9   8F   8D   Ø7   17   2Ø   1F   1F   2C   Ø7   17
Ø31Ø    1Ø   F8   C6   27   DØ   EF   2Ø   54   Ø3   2Ø   7Ø   Ø3   A5   FB   D5   7Ø
Ø32Ø    DØ   D3   A5   FA   D5   8Ø   DØ   CD   A9   Ø4   8D   ØØ   17   EA   EA   EA
Ø33Ø    EA   A9   ØA   85   25   A9   AF   8D   Ø7   17   2Ø   1F   1F   2C   Ø7   17
Ø34Ø    1Ø   F8   C6   25 DØ   EF   2Ø   54   Ø3   C8   4C   3Ø   Ø2   A5   3Ø   48
Ø35Ø    6Ø   37   F1   B8   68   85   3Ø   68   85   31   68   A8   68   AA   A5   31
Ø36Ø    48   A5   3Ø   48   6Ø   B2   8A   AA   98   98   BA   A8   B2   B2   BA   BE
Ø37Ø    68   85   3Ø   68   85   31   8A   48   98   48   A5   31   48   A5   3Ø   48
Ø38Ø    6Ø

XIM
Ø381 36
```

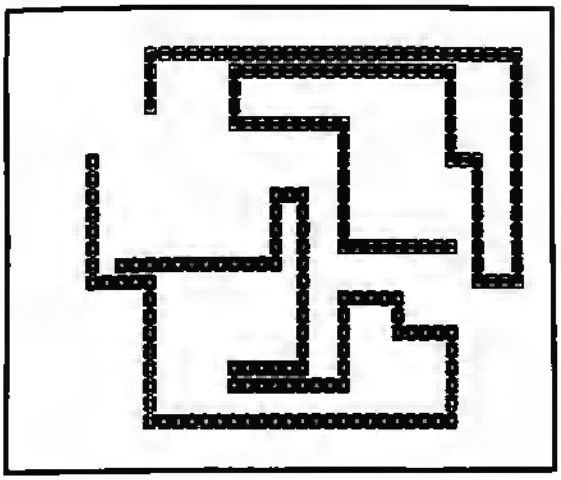
HEX DUMP

Software for the Apple II



SCORE:108 SCORE:105

DYNAMAZE—a dazzling new real-time game. You move in a rectangular game grid, drawing or erasing walls to reflect balls into your goal (or to deflect them from your opponent's goal). Every ball in your goal is worth 100 points, but you lose a point for each unit of elapsed time and another point for each time unit you are moving. Control the speed with a game paddle: play as fast as ice hockey or as slowly and carefully as chess. Back up and replay any time you want to; it's a reversible game. By Don Stone. Integer Basic (plus machine language); 32 K; $9.95.



ULTRA BLOCKADE— the standard against which other versions have to be compared. Enjoy Blockade's superb combination of fast action (don't be the one who crashes) and strategy (the key is accessible open space—maximize yours while minimizing your opponent's). Play against another person or the computer. New high resolution graphics lets you see how you filled in an area—or use reversibility to review a game in slow motion (or at top speed, if that's your style). This is a game that you won't soon get bored with! By Don Stone. Integer Basic (plus machine language); 32 K; $9.95.

What is a REVERSIBLE GAME? You can stop the play at any point, back up and then do an "instant replay", analyzing your strategy. Or back up and resume the game at an earlier point, trying out a different strategy. Reversibility makes learning a challenging new game more fun. And helps you become a skilled player sooner.

WORLD OF ODYSSEY—a new adventure game utilizing the full power of Disk II, which enables the player to explore 353 rooms on 6 different levels full of dragons, dwarfs, orcs, goblins, gold and jewels. Applesoft II 48K; $19.95 includes diskette.

PEROUACKEY—an exciting vocabulary game which pits the player against the clock. The object of the game is to form words from a group of 10 letters which the computer chooses at random. The words must be 3 to 10 characters in length with no more than 5 words of any particular length. Each player has only 3 minutes per turn. The larger the words the higher the score. Applesoft II 16K; $9.95.

APPLESHIP—is a naval game in which two players enter their ships in respective oceans. Players take turns trying to blast their opponent's ships out of the water. The first player to destroy their opponent's ships may win the game. A great low-res graphics game. Applesoft II 32K; $14.95.

Available at your
local computer store

Call or write for our free
SOFTWARE CATALOG

Apple II is a registered
trademark of
Appla Computer, Inc.

DEALER INQUIRIES INVITED

POWERSOFT, INC.

P. O. BOX 157
PITMAN, NEW JERSEY 08071
(609) 589-5500

Programs Available on Diskette
at $5.00 Additional

- Check or Money Order
- Include $1.00 for shipping and handling
- C.O.D. ($1.00 add'tl. charge)
- Master Charge and VISA orders accepted
- New Jersey residents add 5% sales tax

Ask the Doctor

Hints for converting the SYM Tiny PILOT to work on KIM; a Slow Display for the AIM; and, a comparison chart of the AIM, SYM, and KIM expansion pinouts.

Robert M. Tripp
The Computerist, Inc.
P.O. Box 3
S. Chelmsford, MA 01824

"ASK the Doctor" is intended to be a fairly regular column covering matters of interest to the AIM, SYM and KIM users. Parts I through V may be found in issues 9 (Feb '79) through 13 (June '79). Now that the "Doctor is back from vacation", the column will appear fairly regularly again.

This month we have several topics to cover:

Bob Applegate discusses some problems and solutions to using Tiny PILOT on the KIM.

Thomas M. Walsh provides a short program for use with the AIM to slow down the display when using the disassembler.

The Doctor presents a summary of the Expansion and Application pinouts for the AIM, SYM, and KIM along with a description of the KIM-4 Expansion bus structure.

Tiny PILOT for KIM

Machine language programming is very useful for some applications, but for others it is the long way around. Need to print some data? It is possibie, but it is a lot of work. After programming in machine language for a year, I wanted to move up to a high level language such as BASIC. But a BASIC interpreter is not cheap. To make matters worse, most are located from 2000_{16} and up, and my memory ends at $07FF_{16}$. These are two very important facts to consider for any program. I tried writing my own languages but getting a good, small math package was also a major problem. When I saw Tiny Pilot by Nicholas Vrtis (MICRO #16), I was excited! At last I had a neat way to solve some of my programming problems, and to teach some of my non-computer-oriented friends how to program.

Unfortunately, PILOT was written for a SYM, not a KIM. I decided to enter the program, using KIM subroutines in place of SYM subroutines. After entering the program, I started using the interpreter:

```
T: HELLO
S:
@
```

It is a good thing that I don't have a hard-copy terminal because a few feet of paper would have been wasted! Suspecting a mistake in my entry of the interpreter, I checked the program byte-by-byte. Everything was okay. What caused the program to print such garbage? It dawned on me after some thought.

Rereading the last paragraph in Mr. Vrtis' article revealed the answer:

"Tiny PILOT assumes that all registers are preserved by these routines."

Obviously, the KIM monitor does not preserve the registers!

The KIM subroutine OUTCH stores the X register at 00FD, and picks it up again once it is finished. My subroutine SAVOUT (used instead of calls to SYM's OUTCHR) stores the Y register at 00EE, calls OUTCH, reloads the Y register, and exits the routine. SAVIN stores the Y at 00EE, calls GETCH, reloads Y, and exits. SAVCR is a bit longer, because it has to save and restore both registers. It stores Y at the usual place, and X at 00ED. Then it calls CRLF and reloads both registers. Last, but not least, it exits the subroutine.

I located these subroutines in KIM's high RAM, so as to avoid memory problems with Tiny PILOT. Enough room is even left to add a few more statements!

Tiny PILOT is a fun language to use, even if it does have limited capabilities. I hope that some other KIM users will convert between KIM and SYM. I do not know much about SYM's monitor — maybe some MICRO readers could fill me in.

Bob Applegate
Box 148
Bordentown, NJ 08505

Expansion Connector

	Computer			Boards
PIN	AIM	SYM	KIM	MICRO 65
1	SYNC	SYNC	SYNC	**GND**
2	RDY	RDY	RDY	**SYNC**
3	Φ1	Φ1	Φ1	**RDY**
4	*IRQ*	*IRQ*	*IRQ*	*IRQ*
5	S.O.	S.O.	S.O.	S.O.
6	*NMI*	*NMI*	*NMI*	*NMI*
7	*RES*	*RES*	*RES*	*RES*
8	DB7	DB7	DB7	DB7
9	DB6	DB6	DB6	DB6
10	DB5	DB5	DB5	DB5
11	DB4	DB4	DB4	DB4
12	DB3	DB3	DB3	DB3
13	DB2	DB2	DB2	DB2
14	DB1	DB1	DB1	DB1
15	DB0	DB0	DB0	DB0
16	**−12V**	***CS18***	**K6**	**NC**
17	**+12V**	**DBOUT**	***SSTOUT***	**NC**
18	***CS8***	***POR***	**NC**	**DMA**
19	***CS9***	**NC**	**NC**	**+8V**
20	***CSA***	**NC**	**NC**	**+8V**
21	+5V	+5V	+5V	+5V
22	GND	GND	GND	GND
A	AB0	AB0	AB0	**GND**
B	AB1	AB1	AB1	**AB0**
C	AB2	AB2	AB2	**AB1**
D	AB3	AB3	AB3	**AB2**
E	AB4	AB4	AB4	**AB3**
F	AB5	AB5	AB5	**AB4**
H	AB6	AB6	AB6	**AB5**
J	AB7	AB7	AB7	**AB6**
K	AB8	AB8	AB8	**AB7**
L	AB9	AB9	AB9	**AB8**
M	AB10	AB10	AB10	**AB9**
N	AB11	AB11	AB11	**AB10**
P	AB12	AB12	AB12	**AB11**
R	AB13	AB13	AB13	**AB12**
S	AB14	AB14	AB14	**AB13**
T	AB15	AB15	AB15	**AB14**
U	Φ2	Φ2	Φ2	**AB15**
V	R/W	R/W	R/W	Φ2
W	*R/W*	*R/W*	*R/W*	**R/W**
X	TEST	TEST	TEST	***Φ2***
Y	Φ2	Φ2	Φ2	**+5V**
Z	Ram R/W	Ram R/W	Ram R/W	**GND**

Application Connector

	Computer			Boards
PIN	AIM	SYM	KIM	MICRO 65
1	GND	GND	GND	**NC**
2	PA3	PA3	PA3	**NC**
3	PA2	PA2	PA2	**NC.**
4	PA1	PA1	PA1	**NC**
5	PA4	PA4	PA4	**NC**
6	PA5	PA5	PA5	**NC**
7	PA6	PA6	PA6	**NC**
8	PA7	PA7	PA7	**NC**
9	PB0	PB0	PB0	**NC**
10	PB1	PB1	PB1	**NC**
11	PB2	PB2	PB2	**NC**
12	PB3	PB3	PB3	**NC**
13	PB4	PB4	PB4	**NC**
14	PA0	PA0	PA0	**NC**
15	PB7	PB7	PB7	**NC**
16	PB5	PB5	PB5	**NC**
17	**PB6**	**Row 0**	**Row 0**	**NC**
18	**CB1**	**Col F**	**Col F**	**NC**
19	**CB2**	**Col B**	**Col B**	**NC**
20	**CA1**	**Col E**	**Col E**	**NC**
21	**CA2**	**Col A**	**Col A**	**NC**
22	**NC**	**Col D**	**Col D**	**NC**
A	+5V	+5V	.+5V	**NC**
B	**NC**	***CS 00***	**K0**	**NC**
C	**Φ2**	***CS 04***	**K1**	**NC**
D	**R/W**	***CS 08***	**K2**	**NC**
E	**Tape 1B-R**	***CS 0C***	**K3**	**NC**
F	**Tape 1B**	***CS 10***	**K4**	**NC**
H	**Tape 2B-R**	***CS 14***	**K5**	**NC**
J	**Tape 2B**	***CS 1C***	**K7**	**NC**
K	**NC**	***CS 18***	***Decode***	**NC**
L	Audio In	Audio In	Audio In	**NC**
M	Audio Lo	Audio Lo	Audio Lo	**NC**
N	+12V	**RCN-1**	+12V	**NC**
P	Audio Hi	Audio Hi	Audio Hi	**NC**
R	KBD Rtn	KBD Rtn	KBD Rtn	**NC**
S	PTR Rtn	PTR Rtn	PTR Rtn	**NC**
T	KBRD	KBRD	KBRD	**NC**
U	PTR	PTR	PTR	**NC**
V	**Tape 2A**	**Row 3**	**Row 3**	**NC**
W	**Tape 1A**	**Col G**	**Col G**	**NC**
X	**NC**	**Row 2**	**Row 2**	**NC**
Y	***Serial In***	**Col C**	**Col C**	**NC**
Z	**NC**	**Row 1**	**Row 1**	**NC**

Notes: Signals which are the same are in regular type face.
Signals which are different are in **bold** type face.
See your computer manual for a definition of the signals.
The **MICRO 65** bus is identical to the **KIM-4** bus.

Notes: the connections for the application connector are not defined for the **MICRO 65** bus. The application connections are defined by the specific requirements of the expansion board and are generally **not** connected to the host computer.

Slow Down the AIM Display

This program uses AIM subroutines to slow down the display and allows the user to scan thru a disassembly, checking entries made. Holding down the space bar will stop the display at the currewnt display, jsut as at normal speed, but much more controllably.

After the program is entered into RAM, it is activated by pressing the User F-2 key for Slow Display or the User F-1 key for Normal Speed Display. The User F-3 key is unused and is available for other purposes.

The A, Y, and X registers are pushed onto the stack at 0000 thru 0004. At 0005 and 0008, a JSR is made to the AIM Delay subroutine at EC0F, after which X, Y, and A are pulled from the stack and a JMP is made to the Normal Display entry at EF05.

The two small sections at 0013 and 001E are used to reset the addresss which the Monitor points to as the Display Routine: A406,A407. The first subroutine resets the address to Normal Speed, the second sets the address to the Delay routine described above at address 0000, and resets the counter at A417,A418 to FFFF. To speed up the Slow Display, change the value at 0026 to a smaller number, or at address 0005 or 0008 change one of the JSR's to the Delay routine to a NOP.

```
<K>*=0
/25
 0000 48 PHA
 0001 98 TYA
 0002 48 PHA
 0003 8A TXA
 0004 48 PHA
 0005 20 JSR EC0F
 0008 20 JSR EC0F
 000B 68 PLA
 000C AA TAX
 000D 68 PLA
 000E A8 TAY
 000F 68 PLA
 0010 4C JMP EF05
 0013 A9 LDA #05
 0015 8D STA A406
 0018 A9 LDA #EF
 001A 8D STA A407
 001D 60 RTS
 001E A9 LDA #00
 0020 8D STA A406
 0023 8D STA A407
 0026 A9 LDA #FF
 0028 8D STA A417
 002B 8D STA A418
 002E 60 RTS

 K>*=010C
/2
 010C 4C JMP 0013
 010F 4C JMP 001E
```

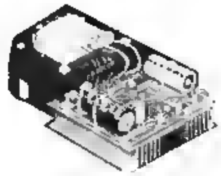
Thomas M. Walsh
5370 Shafter Avenue
Oakland, CA 94618

AIM, SYM, KIM Pinout Summary

One of the features of the AIM, SYM and KIM that make them so compatible is the similarity of their Expansion and Application Connectors. This similarity makes is possible to use a variety of expansion boards: RAM memory, ROM memory, Video, etc., with any one of the three systems. There are some minor differences in the Expansion Connectors, particularly where the KIM did not define a pinout. There are major differences in the Application Connector.

When MOS Technology, developers of the 6502 and the KIM-1, designed their first expansion board, they chose to move all of the Address lines and few other lines to new locations on the Expansion Connector of their new boards. This has been called the KIM-4 Expansion Bus. Since it is used by a number of other manufacturers for expansion boards, and since it serves the AIM and SYM as well as the KIM, I propose to call it the **MICRO 65 Bus**. It is shown in the following chart.

POWER A PLUS™



Designed specifically for the AIM-65
+5V @ 5A and +24V @ 1A
110V/60Hz or 220V/50Hz
Short Circuit protection
Over voltage protection
Enough power to run fully loaded AIM plus several additional boards \$55^{00}

AIM ENCLOSURE™

AIM ENCLOSURE:
A neat, safe package for the AIM 65
Room for one expansion board:
Memory Plus, Video Plus,
or Proto Plus
Enclosure alone: \$45^{00}

AIM PLUS™

With Built-In POWER SUPPLY:
Power A Plus supply
On/off switch with pilot light
Air vents in enclosure to allow heat dissipation
Fully Assembled, Ready to Go \$110^{00}

AIM ALL YOUR NEEDS AT US!!

AIM 65

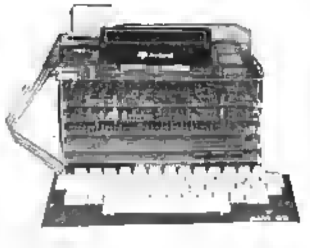
MEMORY PLUS™

Modified for AIM to address at 1000 (4K) boundary - at no charge.
8K Static RAM
Sockets for 8K EPROM
EPROM Programmer
6522 I/O Port
Over 1200 in use. \$200^{00}

VIDEO PLUS™

Glass Teletype Software for AIM is available now and it's FREE. It works with the AIM Monitor and BASIC without any user programming.
UPPER/lower case ASCII
2K Display RAM expandable to 4K
Provision for 2K EPROM
Add up to 128 Programmable Characters by plugging in RAM
Supports ASCII Keyboard
Fully Assembled and Tested: \$245^{00}

CAGE PLUS™



Use with the AIM 65 and MOTHER PLUS
Aluminum frame with card guides
Holds five additional boards with your AIM on top
Package your expanded AIM 65.
\$20^{00} with Mother Plus, \$25^{00} alone.

MOTHER PLUS™

The Sensible Way to Expand:
Fully Buffered and Decoded
Standard Bus Structure
Add up to five expansion boards
Audio/TTY connectors, Port Socket
Fully Assembled and Tested: \$80^{00}

PROTO PLUS™

Add Special Circuits to your system.
Provisions for:
40 14/16 pin sockets
4 24/40 pin sockets
3 voltage regulators
Misc. other components
Two sets of gold plated dual 22 finger connectors with same spacing as AIM.
Wire wrap or solder connections.
Size: 8¼ x 10¾ 40^{00}

Documentation available for MEMORY PLUS (\$10^{00}), VIDEO PLUS (\$10^{00}) and MOTHER PLUS (\$5^{00}). Documentation cost will be applied to purchase price. Prices quoted do not include shipping and handling charges. Please call or write for complete catalog.

THE COMPUTERIST INC

P.O. Box 3
So. Chelmsford, MA 01824
617/256-3649

When ordering, please specify the MEMORY PLUS modification and the VIDEO PLUS Glass Teletype software for your AIM 65 at no charge.

Introducing AppleSeed, our newest publication to whet your Apple* appetite!

We invite you to subscribe to AppleSeed · the magazine that is to the Apple II* what SoftSide is to the TRS-80**. It offers the newest in software programming hints and ideas tailored especially for your computer. AppleSeed features challenging programs for both the do-it-yourselfer and the individual interested in pre-packaged programs and games . . . your own preview of the best available on the market today. A typical slice of AppleSeed consists of one major (new 16K) commercial level program (completely listed for your keying pleasure), accompanied by two or three applications for practical use or fun, supplemented by informative articles to polish your Apple*. Get right to the core of your Apple* needs and order AppleSeed today! 12 issues, 1 year, $15.00. AppleSeed is the newest member of . . .

SoftSide™
PUBLICATIONS
6 South Street, Milford, NH 03055
(603) 673-5144

*A registered trademark of Apple Computers. **A registered trademark of Radio Shack and Tandy Corp.

GREAT PET SOFTWARE

"Precise, humanized, well documented an excellent value" are the applauds now being given to United Software's line of software. These are sophisticated programs designed to meet the most stringent needs of individuals and business professionals. Every package is fully documented and includes easy to understand operator instructions.

DATABASE MANAGEMENT SYSTEM - A comprehensive, interactive system like those run on mainframes! Six modules comprising 42K of programming allow you to; create, edit, delete, display, print, sort, merge, etc., etc. - databases of up to 10,000 records. Printer routines automatically generate reports and labels on demand. 60 pages of concise documentation are included. Requirements - 16-32K PET and 2040 Dual Disk (printer optional). . . .**Cost $125**

ACCOUNTS RECEIVABLE/PAYABLE - A complete, yet simple to use accounting system designed with the small businessman in mind. The United Software system generates and tracks purchase orders and invoices all the way through posting "controlled" accounts payable and accounts receivable subsystems.
Keyed Random Access file methods makes data access almost instantaneous. The low-cost solution for the first time computer user with up to 500 active accounts. Requirements - 32K PET, Dual Disk, any 80-column printer. . . .**Cost $175**

CASH RECEIPTS & DISBURSEMENTS - Makes it a breeze to track all outgoing payments made by any type of business operation. Checks are tracked by number and categorized by type of expense. Sorting, summary, and audit trails make it easy to post to general ledger. This system also categorizes incoming receipts. Uses KRAM file access method. Requirements - 32K PET, Dual Disk (printer optional). . . .**Cost $99.95**

KRAM - Keyed Random Access Method - The new, ultra-fast access method for the PET Disk, provides keyed retrieval/storage of data, in either direct or sequential mode, by either full or partial key values. Written by United Software in 6502 machine code, and designed with the PET in mind, it exploits all the benefits of the PET Disk, allowing full optimization of your system. Eliminates the need for "Sort" routines! KRAM provides flexibility never seen on a micro before. KRAM is modeled after a very powerful access method used on large-scale IBM Virtual Storage mainframes. So "KRAM" all you can into your PET - it will love you for it. . . .**Cost $79.95**

(Sublicenses available to software houses.)

PROGRAMS FOR ENTERTAINMENT	
Space Intruders ("Best Game of 1979")	$19.95
Jury/Hostage	12.50
Kentucky Derby/Roulette	9.95
Alien I.Q./Tank	9.95
Tunnelvision/Maze Chase	14.95
Submarine Attack	9.95
Battle of Midway	7.95
Laser Tank Battle	9.95
Swarm	14.95
Super Startrek	14.95
PET Music Box	29.95

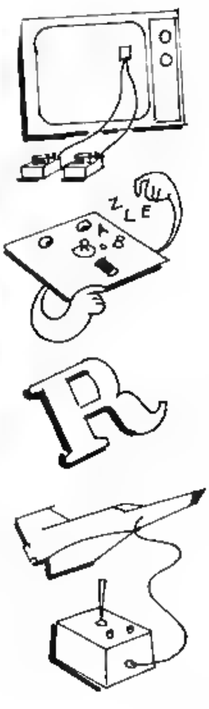
UNITED SOFTWARE PROGRAMS FOR BUSINESS	
Checkbook	$15.95
Mortgage	15.95
Finance	12.95
Bonds	12.95
Stock Analyzer	22.95
Stock Options	24.95
6502 Macro Assembler	49.95

Look for the RED-WHITE-BLUE United Software Display at your local computer dealer, or send check or moneyorder, plus $1.00 shipping to:

UNITED SOFTWARE OF AMERICA
750 Third Ave.
New York, N.Y. 10017

Dealer inquiries invited

APPLE II SOFTWARE



CURSOR PILOT

gives any Apple II game-paddle control of the video cursor. Activate by touching 'ESC', then edit or copy with game-paddle. Supports normal keyboard controls, is transparent to your programs.

on cassette . . . **$5.95**

DATA HANDLER

data base management system. Supports infinite data bases on the Apple II disk drive. Structure data to meet your own needs, up to 255 fields per entry. Advanced data processing allows searching and math to generate reports, extensions, and ledgers. Use for inventory, checks, phone numbers, stocks, lab data, etc. Requires 32K & a disk drive.

on diskette with manual . . . **$49.95**

TYPESETTER

a complete HI-RES graphics character generator and editing system. Allows colors, scaling, upper/lower case, inverse, and can HPLOT letters to any point on the screen. Outputs through regular PRINT statements Use it to label graphs, create ad displays, or print lower case. System includes 35 utility programs and character sets. When ordering, specify it for disk or ROM Applesoft. Needs 32K with ROM, 48K with disk.

on diskette with manual . . . **$29.95**

HIRES UTILITY PACK

Why sweat over HI-RES graphics? Shape Generator lets you build graphic shapes with game paddles, see them at all scales, colors, and rotations. Save them to disk, and Shape Adder puts up to 255 shapes together into a table. Utility Subroutines let you position without plotting, find your last plot, and look at the screen to see if a point is on. Requires 16K with Applesoft ROM.

on diskette . . . **$14.95**

AVAILABLE AT YOUR LOCAL DEALER, OR CALL DIRECTLY AT:

ANDROMEDA COMPUTER SYSTEMS
P.O. BOX 19144
GREENSBORO, N.C. 27410
(919) 852-1482

Visa and Mastercharge gladly accepted

Apple II and Applesoft are trade marks of the Apple Computer Company, Inc.

MIGHTY BYTE IS HERE

PROGRAM—APPLE II		COMPANY	M F G LIST
Bowling League Secretary		Mighty Byte	24.95
Lisa Interactive Assembler	(Req 48K & Disk)	Programma Intl.	34.95
Master Catalog	(Req 32K & Disk)	Programma Intl.	29.95
Apple Pie 'Ver 1.0	(Req 32K & Disk)	Programma Intl.	24.95
Dr. Memory	(Req 32K & Disk)	Muse Co.	49.95
Disk Magic	(Req 32K & Disk)	Programma Intl.	24.95
Format 'Ver 1.0	(Req Disk)	Programma Intl.	24.95
3-D Animation		Programma Intl.	24.95
Super Dungeon	(Req 48K & Disk)	Programma Intl.	24.95
U-Draw		Muse Co.	17.95
U-Draw II	(Req 32K)	Muse Co.	39.95
Three Mile Island	(Req 48K)	Muse Co.	39.95
Escape		Muse Co.	12.95
Tank War		Muse Co.	12.95
Phasor Zap		Programma Intl.	15.95
3-D Ducking		Programma Intl.	15.95
Strato Laser		Programma Intl.	15.95
Depth Charge		Programma Intl.	15.95
Super Othello		Programma Intl.	15.95
Canter Downs		Programma Intl.	15.95
Super Star Wars		Programma Intl.	15.95

Many many more programs available.

Hardware Section	
Centronics 730 Printer	995.00
Heuristics H200 Speechlink	259.00
Comprint 912 Printer (Parallel)	660.00
Comprint 912 Printer (Serial)	695.00
Heath WH-14 Printer	895.00
Vinyl loose-leaf Diskette Pages (Pack of 10)	8.50

Visa & Mastercharge accepted
To introduce you to **MIGHTY BYTE** take 15% off any order.

MIGHTY BYTE COMPUTER
P.O.Box 213
HO-HO-KUS, N.J. 07423
(201)445-8256

Graphics and the Challenger C1P, Part 3

Previous articles have discussed fundamentals of the OSI C1P in regards to the polled keyboard and the expanded graphics set. This article shows how to put the pieces together.

William L. Taylor
246 Flora Road
Leavittsburg, OH 44430

In parts one and two of this series we discussed the C1P and some of its features. To be specific, the polled keyboard and the C1P expanded graphics set. An explanation of how to use the polled keyboard and graphics set in some programs written in Basic. The programs that were presented used only one of the many characters that are a part of the 256 characters available in the C1P character generator ROM. This time I would like to continue with the Large Numbers generation and lead up to the twelve hour clock that was promised last time.

Since this is to be a clock program, I will describe this section of the program first. It may seem rather odd to you that the clock maintine program is buried in the program, but this is how the program evolved. Primarily most of the number generating routines were developed first due to the past part of this series. This is not the best way to write a program, but some programs do evolve in this manner.

The clock mainline routine was a separate program and this portion will be described as a single unit that can be used without the large graphic characters for some of the users that do not have the amount of memory required for the whole program. The clock with the numerals is extremely long. It occupies nearly eight K of user memory. For those users that do not have enough memory to run the entire program I hope that you will use the number generating routines in some of your own programs that would require such things as hit scores or other number displays.

Some beginning criteria for a clock must be given at this point. Any clock that has a digital display must have a number set. The number set must have at least a minimum of four digits of display to qualify as a working clock. Also the hours and minutes must be separate entries. That is, we must have a means of separating the hours and minutes. In addition, we must also have a method of setting the clock to the right time before starting the clock. Finally, we must update the time at some interval. This is usually at one-minute or one-second intervals. The clock should also have a period of day indicator, such as AM or PM.

With this in mind, lets examine the clock portion of our main line Basic program routine that is located at Lines 4000 through 4070. This part of the program will be described in detail and the modifications that are reguired to make it independent from the rest of the program will be given. Looking at the beginning of Line 4000 we see that a GOSUB is executed. The subroutine at line 2900 through 3030 is the fast screen erase machine code memory load routine. This machine code routine will be called to clear the screen for every update of the display. The subroutine is used with both versions of the clock. An explanation of the subroutine was given in part two of this series and the reader is referred to this part for a complete description (MICRO 19:61).

When the program returns from the fast screen routine, the clock must be set to the correct time. This is hours, minutes and seconds where you wish for your clock to start. When you hit a carriage return, the clock begins to run and will be updated on the next whole minute. The hours are contained in the variable S. The minutes are contained in the variable R, and the seconds are contained in the variable Z. The variables are at lines 4004, 4006 and 4007. The actual timer for the clock is a FOR-NEXT loop established at lines 4008 and 4010. This loop should be adjusted to insure accurate timing of your clock. To have the clock run faster, decrease the value of the variable I at line 4008. To decrease the clock rate, increase the value of the variable I at line 4008. After the loop at lines 4008 and 4010 has timed out, the program falls through to the next line. At line 4011 the variable Z is checked to see if a complete minute has been reached (Z = 60). If Z does not = 60 then the timing loop is re-established. When Z is egual to 60, or one minute, the minute counter at line 4013 is incremented. Next at line 4014, a GOSUB to line 4030 resets the second counter to zero. At line 4015 a GOSUB to line 4059 will execute the fast screen erase routine and clear the monitor screen. During this subroutine at lines 4059 through 4065, we will go and check to see what numerals are to be displayed from the hours and minutes look-up tables at lines 59 through 390. It is in these tables that the variables S and R (hours and seconds) are determined and an equivalent numerical display is generated on the monitor screen. When the program returns to the clock mainline program at line 4016, the R variable is checked to see if 60 minutes has been reached. It 60 minutes has not been reached as compared at line 4016, then a new pass through the program is executed. If 60 minutes has been reached (R = 60), then the hours counter will be incremented (variable S). Next, at line 4018 a GOSUB to line 4032 will reset the minute counter and the screen is cleared. A new pass through the look-up table is executed and a new time update is displayed on the monitor screen. At line 4019, the S variable or hours is checked to see if 13 hours has elapsed. We must display 12 hours and 59 minutes. If the S. variable does not equal 13, a new pass through the program is executed. It the variable S is equal to 13 or full hours counter, a GOSUB to line 4034 will cause the Z variable to be reset. At line 4035, the R. variable is reset to zero. At line 4036, the hours counter (S variable) is reset and a GOSUB to line 4059 will clear the monitor screen. The display is updated to 1:00 o'clock and a new pass through the program is executed at line 4037. What all this says is that for each minute that the clock runs, there will be a correct time displayed. For every minute, there will be a new time-up date.

As stated before, the clock routine can be used independent of the whole program. The reader can use this explanation of the routine and the separate program in Listing 2 as a separate program. This listing differs from the routine just described in that is uses a PRINT statement to give the user a viewable readout. Also, this program will update the time every second. If you do not have sutticient memory tor the complete numerical clock, please try the smaller version on your C1P.

In the last part ot this series we discussed how the large numerals were generated. In fact, some ot the large numeral routines are included in this article. At this point, we will continue with the graphics generation and discuss how these subroutines are used in the program for our clock. The contents ot Table 1 lists the line numbers of the key subroutines begin. The reason that we tabulate these subroutines instead ot identitying them in the Basic program is the tact that the Rem statements will occupy memory, and we need to conserve in order to fit the program in 8K ot user memory.

Included with this article is a C1P video memory map that shows the complete video memory as related to the monitor screen. This memory map is in decimal. The locations for the large numbers are shown. These digits will appear at these locations on the monitor screen. With this chart and the number subroutines in the program, you can write programs of your own that require any number displays.

Table 1: Numerical Clock routines

Line		
60 to 385	Numerical look up tables	
1000 to 1020	Least significant digit	One
1100 to 1190	Least significant digit	Two
1200 to 1280	Least significant digit	Three
1300 to 1360	Least significant digit	Four
1400 to 1460	Least significant digit	Five
1500 to 1570	Least significant digit	Six
1600 to 1640	Least significant digit	Seven
1700 to 1760	Least significant digit	Eight
1800 to 1890	Least significant digit	Nine
2000 to 2070	Least significant digit	Zero
2900 to 3030	Fat screen ML load routine	
4000 to 4070	Clock main line program	
5000 to 5080	Second most digit	Zero
5100 to 5120	Second most digit	One
5200 to 5230	Second most digit	Two
5300 to 5340	Second most digit	Three
5400 to 5425	Second most digit	Four
5500 to 5535	Second most digit	Five
5600 to 5635	Second most digit	Six
5700 to 5710	Colon separator for hours and minutes	
6000 to 6025	Third most digit	Zero
6100 to 6130	Third most digit	One
6200 to 6235	Third most digit	Two
6300 to 6335	Third most digit	Three
6400 to 6430	Third most digit	Four
6500 to 6535	Third most digit	Five
6600 to 6645	Third most digit	Six
6700 to 6720	Third most digit	Seven
6800 to 6835	Third most digit	Eight
6900 to 6935	Third most digit	Nine
7000 to 7010	Most Significant digit	One

Table 2: Alarm option program changes

```
2 X = 63232
3 POKE X +1,0: POKE X+3,0: POKE X,255: POKE x+2,0
4 POKE X+1,4: POKE X+3,4
5 POKE X,0
6 GOSUB 4000
4003 INPUT ''SET ALARM''; B,C: D=C+2
4010 NEXT I
4011 Z=Z+1: GOSUB 8007
4063 GOSUB 8005
8000 REM ALARM TEST
8005 IF B=S AND C=R THEN POKE X,1
8006 RETURN
8007 REM TURN OFF ALARM PRESS 1 KEY
8008 G=57088
8009 POKE 530,1
8010 POKE G, 127
8015 IF PEEK (G)=127 THEN POKE X,0
8020 POKE 530,0
8025 RETURN
```

It must be explained at this point that there are subroutines that generate the Least Significant Digits 0 through 9; the Second Most Digits 0 through 6; the Third Most Digits 0 through 9, and finally, the Most Significant Digit 1. The combination of these subroutines together will generate a display of the time. As an example, say the time 12:30 was contained in the S and R variables, we would need to generate digits for four characters. These would be the Most Significant digit one; the Third Most digit two; the Second Most Digit three; and finally, the Least Significant Digit zero. If the variable S contained 12 and the variable R contained 30, when the program goes through to look up tables, variable R would be compared to 30. When 30 was found at Line 215, a GOSUB to Lines 2000 and 5300 would result in the generation of a Second Most digit 3 and a Least Significant digit 0 to be displayed on the screen. Also, when the value for the variable S is found in the look-up table at Line 385, a GOSUB to Lines 6200 and 7000 will cause the generation and display of the Most Significant digit 1 and the Third Most digit 2. From the example, it can be seen that when we are generating a digit display there are usually more than one of the subroutines used to create the graphics.

In the last part of this series, I explained how one example subroutine worked to generate a large number graphic display. The demonstration program in the last part of this series contained subroutines to generate the Least Significant Digits that are a part of this article. Although I described one subroutine in the last part, I will give a description of how one of the subroutines works in this article. The reader may not have the last issue that contained the article, so a description of the number subroutines will make this article a complete entry.

Lets take one subroutine that is used to generate the large numerals and briefly describe its operation. Take the graphics character that represents the numeral 1 in the Least Significant digit location. This subroutine is located at Line 1000 through 1020. First, we must define the locations on our C1P monitor screen that we wish to start to place our character. In the subroutine we are using, the variable A as the video memory pointer. You can see that variable A was defined as video memory locations 54000 to 54128 decimal. This sets up our boundaries in video memory where we wish to place our character. This statement forms part of a FOR-NEXT loop that will be used to load the character that creates the display on the monitor screen. Also note in the statement at Line 1000 we have used a function called the STEP function. This function in a statement will cause the variable to be incremented by the amount contained in the STEP value. In this instance we wish to increment the A variable by 32 for each pass through the loop in the statement line. At the next statement line, the decimal equivalent of a white square will be placed at decimal location 54000. This will be the first part of the data in video RAM that will make up our number character. At the next statement line the program returns to the first line where our FOR-NEXT loop began.

The A variable will be incremented by 32, and the program will fall through the loop again. At the next statement line another square will be placed in video RAM and displayed on the monitor screen. This process will continue until the A variable has been incremented to the final value set in line 1000. This is 54128 decimal. We will now have the graphics representation of the numeral 1 displayed on the monitor screen. With this explanation of the subroutine for the graphics figure 1, you should be able to analyze the remainder of the subroutines to understand them more clearly.

I have written the program to display the large numerals near the bottom left corner of the C1P's monitor screen. If the user should wish these characters displayed at a different location, they can be relocated. This is not a simple task but can be done with the aid of the video memory map that is included as part of this article. From the memory map determine the locations where you wish to have the characters displayed and change the decimal addresses to correspond to the new locations. If you are going to use the number routines for other programs, this may be necessary; but with the clock program as written, remember that the fast screen erase routine will clear only the bottom half of the monitor screen. If you relocate the graphics characters, you will need to have your fast screen erase routine clear the location where you have located your display.

This program is written in subroutines as stated before. In addition to the separate clock and subroutines for the numbers, the fast screen erase routine can be used in other programs that may require this feature. This could be for a rapid screen erase for animated games. The subroutines have many usages even if you cannot run the entire program on your machine.

Basically, this article was written for an OSI Challenger C1P; but the programs will run on other OSI computers with some changes. I have not included these changes in this article because OSI systems are somewhat different. If you have BASIC, you can modify the program to suit your video output such as the 540 in the C2-4P. In addition, a separate listing for an alarm option is included for those users who should have a PIA port in their Challengers. Please refer to Table 2 for the list of the program changes required for the alarm option. The user will need a tone device to implement this option. The alarm option uses a 6820 PIA located at F700 HEX. The A side of the port is used and PA0 is the specific port.

When using either version of the clock, the user must set memory size to protect the machine code routine that is stored in user memory. When using the complete graphics and clock program, the user must set memory size to 8167. When using the shortened version, set memory size to 3840 decimal. When using the clock for either version, the clock timing loop will have to be adjusted for your system to insure accuracy. The clock is tied to the Challenger Processor clock and differs depending on the program being used.

In conclusion, although the BASIC clock requires much memory and will not have the accuracy of a hundred dollar quartz watch, it can be a fine demonstrator. The primary purpose of this article was to describe the C1P's features and teach some programming techniques that could be used by the readers for other programs. This article and programs cover many of the features of BASIC and the Challenger C1P in general. I hope that I have helped some readers and users of the OSI C1P and other OSI systems to grasp a better understanding of BASIC and the graphics capabilities of these fine machines. In the next part of this series, I will show how to do some plotting and create some animated characters using BASIC. Until then, good luck!!

Classified Ads

```
9 GOTO 4000
11 GOSUB 2900
30 GOSUB 4059
59 GOSUB 5700
60 IF R=0 THEN GOSUB 2000:GOSUB 5000
70 IF R=1 THEN GOSUB 1000:GOSUB 5000
80 IF R=2 THEN GOSUB 1100:GOSUB 5000
90 IF R=3 THEN GOSUB 1200:GOSUB 5000
100 IF R=4 THEN GOSUB 1300:GOSUB 5000
110 IF R=5 THEN GOSUB 1400:GOSUB 5000
120 IF R=6 THEN GOSUB 1500:GOSUB 5000
130 IF R=7 THEN GOSUB 1600:GOSUB 5000
140 IF R=8 THEN GOSUB 1700:GOSUB 5000
150 IF R=9 THEN GOSUB1800:GOSUB 5000
160 IF R=10 THEN GOSUB 5100:GOSUB 2000
165 IF R=11 THEN GOSUB 5100:GOSUB 1000
175 IF R=12 THEN GOSUB 1100:GOSUB 5100
190 IF R=13 THEN GOSUB 1200:GOSUB 5100
192 IF R=14 THEN GOSUB 1300:GOSUB 5100
194 IF R=15 THEN GOSUB 1400:GOSUB 5100
196 IF R=16 THEN GOSUB 1500:GOSUB 5100
197 IF R=17 THEN GOSUB 1600:GOSUB 5100
200 IF R=18 THEN GOSUB 1700:GOSUB 5100
202 IF R=19 THEN GOSUB 1800:GOSUB 5100
203 IF R=20 THEN GOSUB 2000:GOSUB 5200
205 IF R=21 THEN GOSUB 1000:GOSUB 5200
207 IF R=22 THEN GOSUB 1100:GOSUB 5200
208 IF R=23 THEN GOSUB 1200:GOSUB 5200
209 IF R=24 THEN GOSUB 1300:GOSUB 5200
210 IF R=25 THEN GOSUB 1400:GOSUB 5200
211 IF R=26 THEN GOSUB 1500:GOSUB 5200
212 IF R=27 THEN GOSUB 1600:GOSUB 5200
213 IF R=28 THEN GOSUB 1700: GOSUB 5200
214 IF R=29 THEN GOSUB 1800:GOSUB 5200
215 IF R=30 THEN GOSUB 2000:GOSUB 5300
216 IF R=31 THEN GOSUB 1000:GOSUB 5300
217 IF R=32 THEN GOSUB 1100:GOSUB 5300
218 IF R=33 THEN GOSUB 1200:GOSUB 5300
219 IF R=34 THEN GOSUB 1300:GOSUB 5300
```

```
220 IF R=35 THEN GOSUB 1400:GOSUB 5300
221 IF R=36 THEN GOSUB 1500:GOSUB 5300
222 IF R=37 THEN GOSUB 1600:GOSUB 5300
223 IF R=38 THEN GOSUB 1700:GOSUB 5300
224 IF R=39 THEN GOSUB 1800:GOSUB 5300
225 IF R=40 THEN GOSUB 2000:GOSUB 5400
226 IF R=41 THEN GOSUB 1000:GOSUB 5400
227 IF R=42 THEN GOSUB 1100:GOSUB 5400
228 IF R=43 THEN GOSUB 1200:GOSUB 5400
229 IF R=44 THEN GOSUB 1300:GOSUB 5400
230 IF R=45 THEN GOSUB 1400:GOSUB 5400
231 IF R=46 THEN GOSUB 1500:GOSUB 5400
232 IF R=47 THEN GOSUB 1600:GOSUB 5400
233 IF R=49 THEN GOSUB 1800:GOSUB5400
234 IF R=50 THEN GOSUB 2000:GOSUB 5500
235 IF R=51 THEN GOSUB 1000:GOSUB5500
236 IF R=52 THEN GOSUB 1100:GOSUB 5500
237 IF R=53 THEN GOSUB 1200:GOSUB 5500
238 IF R=54 THEN GOSUB 1300:GOSUB 5500
239 IF R=55 THEN GOSUB 1400:GOSUB 5500
240 IF R=56 THEN GOSUB 1500:GOSUB 5500
241 IF R=57 THEN GOSUB 1600:GOSUB 5500
242 IF R=58 THEN GOSUB 1700:GOSUB 5500
243 IF R=59 THEN GOSUB1800:GOSUB 5500
245 IF R=48 THEN GOSUB 1700:GOSUB 5400
246 RETURN
300 IF S=0 THEN GOSUB 6000
310 IF S=1 THEN GOSUB 6100
320 IF S=2 THEN GOSUB 6200
330 IF S=3 THEN GOSUB 6300
340 IF S=4 THEN GOSUB 6400
350 IF S=5 THEN GOSUB 6500
355 IF S=6 THEN GOSUB 6600
360 IF S=7 THEN GOSUB 6700
365 IF S=8 THEN GOSUB 6800
370 IF S=9 THEN GOSUB 6900
375 IF S=10 THEN GOSUB 6000:GOSUB 7000
380 IF S=11 THEN GOSUB 6100:GOSUB 7000
385 IF S=12 THEN GOSUB 6200:GOSUB7000
```

```
390 RETURN
1000 FOR A=54000 TO 54128 STEP 32
1010 POKE A,161:NEXT A
1020 RETURN
1100 FOR A= 54000 TO 54002 STEP 1
1110 POKE A,161:NEXT A
1120 POKE 54034,161
1140 FOR A=54064 TO 54066 STEP 1
1150 POKE A,161:NEXT A
1160 POKE 54096,161
1170 FOR A= 54128 TO 54130 STEP 1
1180 POKE A,161:NEXT A
1190 RETURN
1200 FOR A= 54000 TO 54002 STEP 1
1210 POKE A,161:NEXT A
1220 POKE 54034,161
1230 FOR A=54064 TO 54066 STEP 1
1240 POKE A,161:NEXT A
1250 POKE 54098,161
1260 FOR A=54128 TO 54130 STEP 1
1270 POKE A,161: NEXT A
1280 RETURN
1300 FOR A =54000 TO 54064 STEP 32
1310 POKE A , 161: NEXT A
1320 FOR A=54064 TO 54066 STEP 1
1330 POKE A,161:NEXTA
1340 FOR A=54002 TO 54130 STEP 32
1350 POKE A,161: NEXT A
1360 RETURN
1400 FORA=54000 TO 54002 STEP1

1400 FORA=54000 TO 54002 STEP1
1410 POKE A,161: NEXT A
1420 FOR A=54064 TO 54066 STEP 1
1425 POKE A,161:NEXT A
1430 FOR A=54128 TO 54130 STEP 1
1440 POKE A,161: NEXT A
1450 POKE 54032,161: POKE 54098,161
1460 RETURN
1500 FOR A=54000TO54002 STEP 1
1510 POKE A, 161: NEXT A
1520 FOR A=54064 TO 54066 STEP 1
1530 POKE A, 161: NEXT A
1540 FOR A=54128 TO 54130 STEP 1
1550 POKE A, 161:NEXT A
1560 POKE 54032,161:POKE 54096,161:
        POKE 54098,161
1570 RETURN

1600 FOR A=54000 TO 54002 STEP 1
1610 POKE A, 161: NEXT A
1620 FOR A=54002 TO 54130 STEP 32
1630 POKE A,161:NEXT A
1640 RETURN
```

```
1700 FOR A=54000 TO 54128 STEP 32
1710 POKE A,161:NEXT A
1720 FOR A= 54002 TO 54130 STEP 32
1730 POKE A, 161: NEXT A
1740 FOR A=54001 TO 54129 STEP 64
1750 POKE A, 161: NEXT A
1760 RETURN
1800 FOR A=54002 TO 54130 STEP 32
1810 POKE A, 161: NEXT A
1820 FOR A= 54000 TO 54002 STEP 1
1830 POKE A, 161: NEXT A
1840 FOR A =54064 TO 54066 STEP 1
1850 POKE A, 161: NEXT A
1860 FOR A=54128 TO 54130 STEP 1
1870 POKE A, 161: NEXT A
1880 POKE 54032, 161
1890 RETURN
1900 FOR A=53998 TO 54126 STEP 32
1910 POKE A, 161: NEXT A
1930 RETURN
2000 FOR A= 54000 TO 54002 STEP 1
2010 POKE A,161:NEXT A
2020 FOR A=54000 TO 54128 STEP 32
2030 POKE A, 161: NEXT A
2040 FOR A= 54002 TO 54130 STEP 32
2050 POKE A, 161:NEXT A
2060 POKE 54129, 161
2070 RETURN
2900 FOR R=8168 TO 8191
2920 READ F:POKE R,F:NEXT R
2925 RESTORE
2930 RETURN
3000 DATA 169,32,160,4,162,0,157,0
3010 DATA 210,232,208,250,238,240
3020 DATA 31,136,208,244,169,210
3030 DATA 141,240,31,96

4000 GOSUB 2900
4002 PRINT" TIME HRS SEC MIN"
4004 INPUT S
4006 INPUT R
4007 INPUT Z
4008 FOR I=1 TO 725
4010 NEXT I
4011 Z=Z+1
4012 IF Z<60 THEN 4008
4013 IF Z=60 THEN R=R+1
4014 IF Z=60 THEN GOSUB 4030
4015 GOSUB 4059
4016 IF R<60 THEN GOTO 4008
4017 IF R=60 THEN S=S+1
4018 IF R=60 THEN GOSUB 4032
4019 IF S<13 THEN 4008
4020 IF S=13 THEN 4034
```

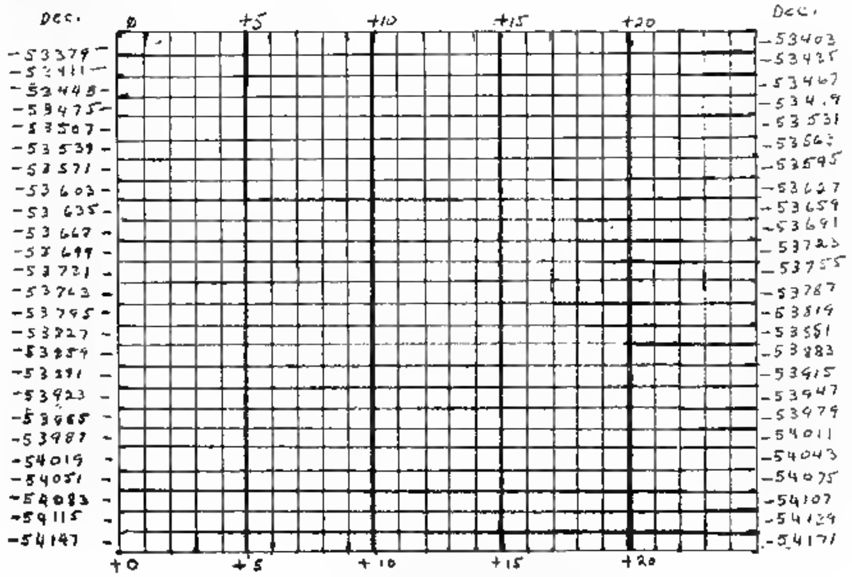
```
4030 Z=0
4031 RETURN
4032 R=0:GOSUB 4059
4033 RETURN
4034 Z=0
4035 R=0
4036 S=1:GOSUB 4059
4037 GOTO 4008
4053 POKE 11,232:POKE 12,31
4054 GOTO 5
4059 POKE 11,232:POKE 12,31:X=USR(X)
4060 GOSUB 300
4062 GOSUB 59
4065 RETURN
5000 FOR A=53996 TO 53998 STEP 1
5010 POKE A,161:NEXT A
5020 FOR A=54124 TO 54126 STEP 1
5030 POKE A,161:NEXT A
5040 FOR A=53996 TO 54124 STEP 32
5050 POKE A,161:NEXT A
5060 FOR A=53998 TO 54126 STEP 32
5070 POKE A,161:NEXT A
5080 RETURN
5100 FOR A=53998 TO 54126 STEP 32
5110 POKE A,161:NEXT A
5120 RETURN
5200 FOR A=53996 TO 53998 STEP 1
5205 POKE A,161:NEXT A
5210 FOR A=54124 TO 54126 STEP 1
5215 POKE A,161:NEXT A
5220 POKE 54092,161:POKE54030,161
5230 FOR A=54060 TO 54062 STEP 1
5235 POKE A,161:NEXT A
5240 RETURN
5300 FOR A=53996 TO53998 STEP 1
5300 FOR A=53996 TO53998 STEP 1
5305 POKE A,161:NEXT A
5310 FOR A=54060 TO 54062 STEP 1
5315 POKE A,161:NEXT A
5320 FOR A=54124 TO 54126 STEP 1
5325 POKE A,161:NEXT A
5330 POKE 54030,161:POKE 54094,161
5340 RETURN
5400 FOR A=53998 TO 54126 STEP 32
5405 POKE A,161:NEXT A
5410 FOR A=53996  TO 54060 STEP 32
5415 POKEA,161:NEXT A
5420 POKE 54061,161
5425 RETURN
5500 FOR A=53996 TO 53998 STEP 1
5505 POKE A,161:NEXT A
5510 FOR A=54124 TO 54126 STEP 1
5515 POKE A,161:NEXT A
5520 FOR A=54060 TO 54062 STEP 1
5525 POKE A,161:NEXT A
```

```
5530 POKE 54028,1 61:POKE 54062,
          161:POKE 54094,161
5535 RETURN
5600 FOR A=53996 TO 53998 STEP 1
5605 POKE A,161:NEXT A
5610 FOR A=54060 TO 54062 STEP 1
5615 POKE A,161:NEXT A
5620 FOR A=54124 TO 54126 STEP 1
5625 POKE A,161:NEXT A
5630 POKE 54092,161:POKE54094,
          161:POKE 54028,161
5635 RETURN
5700 POKE 54027,172:POKE54091,172
5710 RETURN
6000 FOR A=53992 TO 54120 STEP 32
6005 POKE A,161:NEXT A
6010 FOR A=53994 TO 54122 STEP 32
6016 POKE A,161:NEXT A
6020 POKE 53993,161:POKE 54121,161
6025 RETURN
6100 FOR A=53994 TO 54122 STEP 32
6120 POKE A,161:NEXT A
6130 RETURN
6200 FOR A=53992 TO 53994
6205 POKE A,161:NEXT A
6210 FOR A=54056 TO 54058
6215 POKE A,161:NEXT A
6220 FOR A=54120 TO 54122
6225 POKE A,161:NEXT A
6230 POKE 54026,161:POKE 54088,
          161:POKE54057,161
6235 RETURN
6300 FOR A=53992 TO 53994
6305 POKE A,161:NEXT A
6310 FOR A=54056 TO 54058
6315 POKE A,161:NEXT A
6320 FOR A=54120 TO 54122
6325 POKE A,161:NEXT A
6330 POKE 54026,161:POKE 54090,161
6335 RETURN
6400 FOR A=53994 TO 54122 STEP 32
6405 POKE A,161:NEXT A
6410 FOR A=54056 TO 54058
6415 POKE A,161:NEXT A
6420 FOR A=53992 TO 54056 STEP 32
6425 POKE A,161:NEXT A
6430 RETURN
6500 FOR A=53992 TO 53994
6505 POKE A,161:NEXT A
6510 FOR A=54056 TO 54058
6515 POKE A,161:NEXT
6520 FOR A=54120 TO 54122
6525 POKE A,161:NEXT A
6530 POKE 54024,161:POKE 54098,161
```

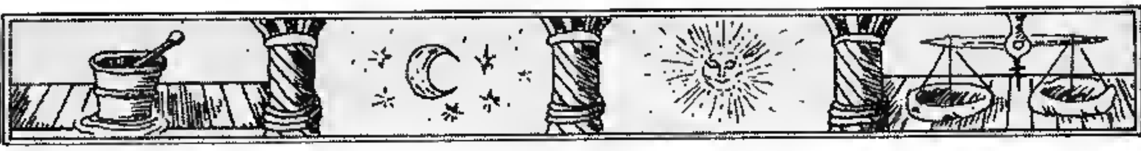
```
6535 RETURN
6600 FOR A=53992 TO 54120 STEP 32
6605 POKE A,161:NEXT A
6610 FOR A=53992 TO 53994
6615 POKE A,161:NEXT A
6620 FOR A=54120 TO 54122
6625 POKE A,161:NEXT A
6630 FOR A=54056 TO 54058
6635 POKE A,161:NEXT A
6640 POKE 54090,161
6645 RETURN
6700 FOR A=53992 TO 53994
6705 POKE A,161:NEXT A
6710 FOR A=53994 TO 54122 STEP 32
6715 POKE A,161:NEXT A
6720 RETURN
6800 FOR A=53994 TO 54122 STEP 32
6805 POKE A,161:NEXT A
6810 FOR A=53992 TO 54120 STEP 32
6815 POKE A,161:NEXT A
6820 POKE 53993,161:POKE 54057,
          161:POKE 54121,161
6825 RETURN
6900 FOR A=53994 TO 54122 STEP 32
6905 POKE A,161:NEXT A
6910 FOR A=54056 TO 54058
6915 POKE A,161:NEXT A
6920 FOR A=53992 TO 53994
6925 POKE A,161:NEXT A
6930 POKE 54024,161
6932 FOR A=54122 TO 54120 STEP -1
6934 POKE A,161 :NEXT A
6935 RETURN
7000 FOR A=53990 TO 54118 STEP 32
7005 POKE A,161:NEXT A
7010 RETURN
```

```
2 PRINT" ENTER TIME HRS MIN SEC"
3 GOTO 50
4 INPUT C
6 INPUT B
7 INPUT A
8 FOR I=1 TO 450
10 NEXT I
11 GOSUB 60
12 A=A+1
13 IF A<60 THEN GOTO 8
14 IF A=60 THEN B=B+1
15 IF A=60 THEN GOSUB 30
16 IF B<60 THEN GOTO 8
17 IF B=60 THEN C=C+1
18 IF B=60 THEN GOSUB 32
19 IF C<13 THEN GOTO 8
20 IF C=13 THEN GOTO 34
30 A=0
31 RETURN
32 B=0
33 RETURN
34 A=0
35 B=0
36 C=1
37 GOTO 8
49 REM SET MEMORY SIZE TO 4050
50 FOR Q=4072 TO 4095
51 READ N:POKE Q,N
52 NEXT Q
53 POKE 11,232:POKE 12,15
54 GOTO 4
60 X=USR (X)
61 PRINT" TIME";C;":";B;".";A
62 RETURN
70 DATA 169,32,160,8,162,0,157,0
75 DATA 208,232,208,250,238,240
80 DATA 15,136,208,244,169,208
85 DATA 141,240,15,96
```

C1P Memory Map in decimal 25 x 25 format



DDJ



DR. DOBB'S JOURNAL
of COMPUTER Calisthenics & Orthodontia

A REFERENCE JOURNAL FOR USERS OF SMALL COMPUTERS

For those who want their computer information straight, sweet and to the point, there is no substitute: there is only *Dr. Dobb's Journal.*

Dr. Dobb's Journal is a highly respected reference journal which fills a unique and solid niche within the microcomputing world. We publish discussions and examples of general purpose system tools, articles on legislation or trends affecting computerists, a consumer watchdog column, as well as a variety of monthly columns designed to guide and help readers wend their way through the confusion of a new and burgeoning industry.

Recent issues have included:

Discovering the 26th Prime • tiny 8ASIC for the F8 • Some Improvements on 8080 Pilot • Page List for the Apple • On Computer Crime Bill S240 • An Electromechanical Household Servant • Producing Pictures on your Computer with a Diablo Printer •

OUR READERS SAY:

"Your level of information is fantastic . . . the best!"

"*Dr. Dobb's* has presented some of the best and most intelligent software for the advanced hobbyist. Please, please stay that way!"

"I received my introduction to your remarkable journal and was as pleased as a Lovecraft wizard coming upon a copy of the *Necronomicon.* Really, yours is the only real computer journal going. *Dr. Dobb's* is to the other "small systems" periodicals as *Scientific American* is to *Popular Mechanics.*"

"*Dr. Dobb's Journal* is a truly impressive periodical unlike others I've read. It contains valuable and useful software. I look forward to its delivery every month."

Dr. Dobb's Journal $15/yr., 10 issues.

SUBSCRIPTION ORDER

Dr. Dobb's Journal □ $28 Two Years ($12 off cover price) D4
□ $15 One Year ($5 off cover price)

Please start my subscription to **Dr. Dobb's Journal,** published ten times per year.

NAME ______________________________

ADDRESS ______________________________

CITY/STATE ______________________ ZIP ________

□ Check enclosed □ Bill me (**U.S.** only)

Signature ______________________________

International Rates:

Canada/Mexico *First Class* □$48 2 years
□$25 one year

Rest of World *Airmail* □$64 2 years
□$33 one year

World *Surface Mail* □$38 2 years □$20 one year
I will risk the lengthy and unreliable delivery
of Surface Mail.

(signed) ______________________________

Payment must be in $US drawn on a US bank.

Send this form or a facsimile to:

This offer good until 6/30/80

PEOPLE'S COMPUTER COMPANY 1263 El Camino Real, 8ox E, Menlo Park, California 94025

PCC

The MICRO Software Catalogue: XVII

Mike Rowe
P.O. Box 6502
Chelmsford, MA 01824

Name: **TXT/ED 2.0**
System: **APPLE II**
Memory: **32K RAM with ROM Applesott, or 48K RAM (disk) Applesoft**
Language: **APPLESOET and Machine Language**
Hardware: **APPLE It, Disk II** (A printer with Serial or Parallel Interface is desirable)

Description: TXT/ED is a disk-based Word Processor and an APPLESOFT BASIC program editor. Major features of theTXT/ED 2.0 include: no contusing CONTROL characters within your text, full right margin justification, merging of multiple disk tiles, tind or change any text sequence In text memory, fully supported upper and lower case letters, extensive Text Formatting capabilities (including text tines, page numbers, two column print tormat), full data display (including page scrolling), slow-list and stop-list display of text data, conversion of APPLESOFT programs to text form for editing, then reconversion back to run-time tormat, selective saving of all or part of text memory to disk, multiple Disk II fully supported, easy creation ot APPLE DOS 'EXEC' files, up to nine Tab Stops may be set for columnar data, line or paragraph block move, duplicate and center. Easy interfacing to any type printer.

Coples: **Just Released**
Price: **$65.00 on disk**
Includes: System disk, 51 page instruction manual
Author: **Gerald H. Rivers**
Available: G.H. Rivers
P.O. Box 833
Madison Heights, MI 48071

Name: **ISAM—DS**
System: **APPLE II**
Memory: **3K plus index table storage**
Language: **Appfesoft**
Hardware: **APPLE II, Disk II**

Description: ISAM—DS is an integrated set of fifteen utility routines that facilitate the creation and manipulation ot indexed files. Records on indexed files may be easily and quickly retrieved, either directly (randomly) or in sequence. Each record is identified by a key data value. The key values do not have to be part of the record; they do not have to be unique to each record; and partial key values may be used in retrieving records. The interface between ISAM—DS and an Applesott program is through a single entry point (GOSUB) and nine variables. Eiles can be created, opened, closed, copied, and erased. Records can be written, read, changed, and deleted. File space that is treed by deleting a record is automatically reused when another record is added. There is never a need to "clean up" a file because ot update activity. ISAM—DS is a must for writing business systems for the APPLE II and is equally useful in personal programs or for learning index-sequential file processing techniques.

Copies: **Just Released**
Price: **$50.00** (Texas residents add 5 percent sales tax.)
Includes: Integrated set ot routines, documentation for the routines, and a sophisticated mailing list program that demonstrates ISAM—DS capabilities. Append routines tor DOS 3.1 and 3.2 are also included. The append routines are used to join the ISAM—DS package to an Applesoft program.
Author: **Robert E. Zant**
Available: Decision Systems
P.O. Box 13006
Denton, TX 76203

Name: **COMMODITY EILE**
System: **APPLE It Computer**
Memory: **2K with Applesoft ROM**
48K with Applesoft RAM
Language: **APPLESOET II**
Hardware: **Disk II, 132 column printer (optional)**

Description: Commodity File stores and retrieves virtually every commodity traded on all Euture's exchanges. A selt-prompting program allowing the user to enter short/long contracts. Computes gross and net protits/losses, and maintains a running cash balance. Takes into account any amending of cash balances such as new deposits or withdrawals from the account. Instantaneous readouts (CRT or printer) ot contracts on file, cash balances, P/L statement. Includes color bar graphs depicting cumulative and individual transactions. Also includes routine to proofread contracts before filing.

Copies: **60plus**
Price: **$19.95** Diskette plus $1.95 P&H, First Class, Check or money order.
Includes: System diskette and full documentation.
Author: **S. Goldstein**
Available: Mind Machine, Inc.
31 Woodhollow Lane
Huntington, N.Y. 11743

Copyrighted: 3/1/79, all rights reserved.

Name:	**Astronomy Software**
System:	**PET**
Memory:	**8K or more**
Language:	**BASIC**

Description: Astronomical programs for PET; Time, coordinate, and compass direction of celestial objects. These and many other programs for PET by JAPS —Jacksonville Area Pet Society.

Copies:	**Hundreds**
Price:	**$1.50 per program,** plus $1.00 for tape and postage.
Includes:	Cassette
Available:	Send self-addressed stamped envelope to: Pet Library 401 Monument Rd. No. 123 Jacksonville, FL 32211

Name:	TRAP65
System:	**Any 6502 based microcomputer**
Memory:	**Not applicable**
Language:	**Not applicable**

Description: TRAP65 is a hardware device which plugs into the 6502 microprocessor's socket. TRAP65 monitors each opcode that the 6502 executes; and if an unimplemented opcode is about to be executed, a BRK Instruction is forced on the data bus. This prevents system crashes especially when debugging machine language programs. TRAP65 can also be used to extend the 6502 instruction set. For example, 0F is an unimplemented opcode that can, via appropriate routine, become a PHX (push X) instruction or any function that you can define in software.

Copies:	**Just released.**
Price:	**$149.95**
Authors:	**J. R. Hall and C. W. Moser**
Available:	Eastern House Software 3239 Linda Drive Winston-Salem, N.C. 27106

Name:	**Applesoft Tape Verifier**
System:	**Apple II or Apple II Plus**
Memory:	**16K RAM**
Language:	**Applesoft**

Description: This program gives the Apple computer the capability of verifying Applesoft programs that have been saved out on tape. It does this without destroying the original program. The program will work with either the Apple II or the Apple II Plus computers and will also work with either RAM or ROM Applesoft.

Copies:	**Just released.**
Price:	**$20.00**
Available:	Softsell Associates 2022 79th Street Brooklyn, N.Y. 11214

Name:	Mailing List Database
System:	**APPLE II**
Memory:	**48K**
Language:	**Applesoft**
Hardware:	**Applesoft on ROM and at least one disk drive.**

Description: This new, user oriented mailing list program introduces professional quality and speed to the processing of name and address files. Labels on printed lists can be readily produced at any time. Features include: single keystroke commands, convenient data entry, machine language searches, machine language sorts, flexible application and versatile output. Mailing List Database is supplied on disk and comes with a program for automatically converting existing text mailing list files. It requires 48K Apple II with Applesoft on Rom (or language card) and at least one disk drive.

Copies:	**Many**
Price:	$34.50 (WA residents add 5.3 percent sales tax).
Authors:	**Robert C. Clardy and Christopher Anson**
Available:	Synergistic Software 5221 - 120th Avenue, S.E. Bellevue, WA 98006

Name:	**Typesetter**
System:	**APPLE II OR APPLE II Plus**
Memory:	**32K**
Language:	**Applesoft II and Machine**
Hardware:	**Disk II**

Description: The Typesetter is a complete HIRES character generating and editing system. It features foreground and background colors, upper/lower case, inverse video, rotated characters, and foreign characters sets (including Greek, Hebrew, and PET graphics). Characters may be positioned anywhere on the screen, eliminating the usual 40X24 grid. The output is through regular print statements. Scale, color, and other functions are implemented using standard Applesoft II commands. Use it to label graphs, create ad displays, or print lower case or foreign languages. A character set editing program is included. Character tables are compatible with Apple's character generator on user contributed Volume 3. The system includes 35 utility programs and character sets plus manual.

Copies:	**30**
Price:	**$24.95**on diskette. Please specify disk or ROM Applesoft. N.C. residents add 4 percent sales tax.
Authors:	**Jeff Schmoyer and Joe Budge**
Available:	ANDROMEDA COMPUTER SYSTEMS P.O.Box 19144 Greensboro, N.C. 27410 (919) 852-1482

Name:	**Morse Code Transceive Program**
System:	**Ohio Scientific C1-P and Superboard 2**
Memory:	**Standard 4K**
Language:	**Machine Language and Basic**
Hardware:	**Decoded Port Required** (schematic supplied)

Description: The program is designed for the HAM that wants a truly useful morse code program. It will copy CW up to 60 WPM. The copy program tracks the incoming code speed and, therefore, the user needs only to set the transmit speed. The program comes up in receive mode and is ready to copy. To go to transmit mode, simply press the spacebar. A cursor will now appear in the upper left hand corner of the screen. This is the position of the character that is presently being sent. As characters are entered from the keyboard they will be displayed across the screen. After each character is sent, the display is updated by a fast machine language routine which moves all the characters over one position. While in transmit mode, the following keys have special meaning:

; (semicolon) returns to receive mode

:(colon) program will ask for a change in code speed.

RUBOUT key will backspace cursor in order to easily make corrections.

Price:	**$14.95 prepaid, M/C and VISA accepted**
Includes:	Program on cassette, loading instructions, operating instructions, detailed instructions and schematic for building required port.
Author:	**Steve Olensky, WB4DCL**
Available:	Olensky Bros., Inc. Computer Sales Division 3763 Airport Blvd. Mobile, AL 36608

Software Catalog Note

Do you have a software package you want publicized? Our Software Catalogue is a good opportunity to receive some free advertisement. This regular feature of MICRO is provided both as a service to our readers and as a service to the 6502 industry which is working hard to develop new and better software products for the 6502 based system. There is no charge for listings in this catalog. All that is required is that material for the listing be submitted in the listing format. All info should be included. We reserve the right to edit and/or reject any submission. Some of the submissions are too long. We might not edit the description the same way you would, so please, be brief and specific.

STOCK MARKET ANALYSIS PROGRAM
DJI WEEKLY AVERAGE 1897-1980

ANA1 (ANALYSIS 1) is a set of BASIC Programs which enables the user to perform analyses on the Dow Jones Industrial weekly average data. From 6 months to 5 years of user selected DJI data can be plotted on the entire screen in one of 5 colors using Apples' High Resolution capabilities. The DJI data can be transformed into different colored graphic representations called transforms. They are: user specified moving averages; a least squares linear fit (best straight line); filters for time, magnitude, or percentage changes; and user created relationships between the DJI data, a transform, or a constant using +,-,x,/ operators Colored lines can be drawn between graphic points. Graphic data values or their dates of occurrence can be displayed in text on the screen. Any graph or text can be outputted to a users printer. The Grid Scale is automatically set to the range of the graphs or can be user changed. As many colored graphs as wanted can be plotted on the screen and cleared at any time. The user can code routines to operate on the DJI/transform data or create his own disk file data base. ANA1 commands can be used with his routines or data base. An Update program allows the user to easily update the DJI file with current DJI weekly data.

The ANA1 two letter user commands are: CA = Calculate, no graph. CG = Clear Graphs, leave Grids. CK = Checking out program, known data. CO = Color of next graph (red, green, violet, white, blue). CS = Clear Screen. DL = Draw Line between points. FI = Filter data for time, magnitude, or percent change. FU = Data, transform, or constant Function with +,-,x,/ operator. GD = Graphic mode, display all Graph Data on screen. GR = Graph data to screen. GS = Set Grid Scale. HE = Help, summary of any commands usage. LD = Load Data from disk file from inputted date to memory. LG = Leave Graphs, automatic Grid rescaling. LO = Look, select a range of the LD data and GR; All commands can now be used on this range. LS = Least squares linear fit of the data. MA = Moving Average of the data. NS = No Scale, next graph on screen does not use Grid Scale. NT = No Trace. PR = User implimented Printer routine. TD = Text mode, display Text Data on screen. TI = Time number to date or vice versa. TR = Trace. TS = Text Stop for number of lines outputted to screen when in TD. U1/U2 = User 1/2 implimented routines. VD = Values of Data outputted in text. VG = Values of Grid: low/high/delta. VT = Values of Transform outputted in text.

APPLE® II, 48 K, APPLESOFT
ROM CARD, DISK II DOS 3.2
ANA1 DISK & MANUAL . . . $49.95
(CA residents add 6% sales tax)

GALAXY
DEPT. MI1
P.O. BOX 22072
SAN DIEGO, CA 92122

YOU BOUGHT THE BEST! NOW LEARN TO USE IT!

AT LAST!

A magazine devoted to *Applications* as well as Technique for the Apple Computer.

THE APPLE SHOPPE WILL TEACH YOU NOW TO DO ALL THOSE FANCY THINGS ON THE APPLE. LEARN HOW OTHERS ARE USING THEIR APPLES IN THE HOME, SCHOOLS AND BUSINESSES.

CHECK THESE FEATURES:

- ✓ Feature Articles on Apple Applications
- ✓ Program of the Month—'How To' with Listings
- ✓ New Products Review—All Boards, Pascal, etc.
- ✓ Language Lab—Learn Basic, Pascal, Forth, Lisp, Pilot
- ✓ Future Projects—Participate in a new program design called "The China Syndrome"
- ✓ Graphics Workshop—Learn secrets formerly known only to 'Super Programmers'

☐ YES I want to learn how to get the most out of my Apple. Send me a one year subscription. I enclose $12.

NAME: ______________________________

ADDRESS: ______________________________

CITY ______________ STATE ______ ZIP ______ PHONE ______

☐ NO, I already know it all, but send me a free sample of next issue

Send check or money order to: Apple Shoppe, P.O. Box 701, Placentia, CA 92670
or call (714) 996-0441

STONEWARE for APPLE II*

For the Serious Business or Home User:
MICRO MEMO

MICRO MEMO is the first sophisticated "Desk Calendar" program to make good use of your computer's power

★ Micro Memo includes one time, weekly, monthly, semi-annual and annual reminders.

★ Monthly reminders may be for fixed or "floating" dates (ex. 1st Saturday of every month).

★ Each reminder allows choice of one week, 2 week or 1 month advance notice—reminds you ahead of time to prepare for meetings, purchase tickets, make reservations, etc.

★ Micro Memo includes "shorthand" for fast memo entry, greater capacity

★ Micro Memo will display or print any day's or week's reminders

★ Micro Memo is a "perpetual" calendar—automatically creates new months with all appropriate memos (birthdays, anniversaries, monthly meetings, etc.) as past months are dropped—system holds full year's reminders on one disk.

★ Micro Memo "knows" most major holidays

★ Supports Mountain Hardware clock (optional)

★ "Bomb Proof" menu driven command and date entry.

★ Requires 48K, disk, RAM or ROM Applesoft.

$39⁹⁵

STONEWARE
Microcomputer Software
P.O. Box 7218, Berkeley, CA 94707
(415) 548-3763

And Just for Fun:
TRANQUILITY BASE

$24⁹⁵

TRANQUILITY BASE is a fast high resolution Lunar Lander game by Bill Budge, creator of Apple's "Penny Arcade." TRANQUILITY BASE is just like the popular arcade game, including multiple moonscopes, craft rotation, and zoom in for a close-up view as you approach the Lunar surface.

TRANQUILITY BASE requires 32K and disk

Available at your favorite computer store or direct from STONEWARE (add $2 shipping & handling; Calif. residents add sales tax. Visa & MasterCharge accepted, no C.O.D.'s).

DEALER INQUIRIES INVITED

*Apple II is a Trademark of Apple Computer, Inc

Bringing Music Home

LET MICRO MUSIC TURN YOUR APPLE II INTO A FAMILY MUSIC CENTER!

. Sing along
. Compose
. Play
. Learn from Specialists

VISIT THE APPLE DEALER NEAREST YOU AND ASK FOR A DEMONSTRATION OF MMI'S MUSIC COMPOSER ™
The **MUSIC COMPOSER** is an **APPLE II®** compatibile, low-cost music system designed by the folks at MMI. Our music software was designed by leading experts in music education. A simple step-by-step instruction manual leads you through entering, displaying, editing, and playing music with up to four voices—soprano, alto, tenor, and bass. You can change the sound of each voice to reed, brass, string, or organ sounds and you can even color your own music sounds!

HAVE FUN! THE MUSIC COMPOSER comes complete with an instruction manual, software disk or cassette—in either Integer or Applesoft **ROM BASIC,** and the **MICRO MUSIC DAC** music card. Just plug the **MICRO MUSIC DAC** into the **APPLE** extension slot and connect the audio cable to a speaker.

Suggested retail price $220.

Ask your local dealer for information on MMI music software products, or contact:

MMI ™
Micro Music Inc (309) 452-6991
309 Beaufort, University Plaza, Normal, IL 61761

PET Word Processor

8K and 16/32K PET versions

This program permits composing and printing letters, flyers, advertisements, manuscripts, etc., using the COMMODORE PET and a printer.

Printing directives include line length, line spacing, left margin, centering and skip. Edit commands allow you to insert lines, delete lines, move lines and paragraphs, change strings, save files onto and load files from cassette (can be modified for disk), move up, move down, print and type.

Added features for the 16/32K version include string search for editing, keyboard entry during printing for letter salutations, justification, multiple printing and more.

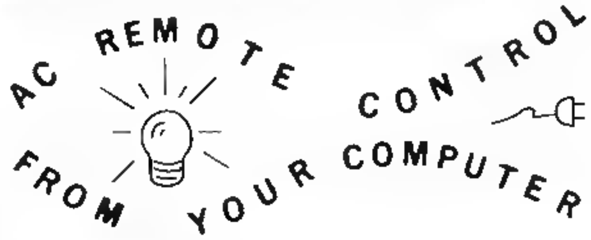
A thirty page instruction manual is included.

The CmC Word Processor Program for the 8K PET is $29.50. The 16/32K version is $39.50.

Order direct or contact your local computer store.

VISA AND M/C ACCEPTED — SEND ACCOUNT NUMBER, EXPIRATION DATE AND SIGN ORDER.
ADD $1 PER ORDER FOR SHIPPING & HANDLING — FOREIGN ORDERS ADD 10% FOR AIR POSTAGE

CONNECTICUT microCOMPUTER, Inc.
150 POCONO ROAD
BROOKFIELD, CONNECTICUT 06804
TEL: (203) 775-9659 TWX: 710-456-0052



TRS-80 PET S100
APPLE KIM AIM65

INEXPENSIVE CONTROL SOLUTION FOR
HOME SECURITY · ENERGY CONSERVATION
GREENHOUSES · ENVIRONMENTAL CONTROL
INDUSTRIAL CONTROL · LABORATORIES

CmC's μDAC system now includes an interface to the BSR X-10 remote control modules. These low-cost modules allow control over lamps, motors and appliances. With the CmC X-10 interface your computer can control 256 separate devices. Lamps can be turned on or off, dimmed or brightened. Alarms, kitchen appliances, hi-fis, TVs, motors, pumps, heaters and more can be put under your computer's control.

Direct plug-in and software for most computers.

Circle the reader service number, call or write for our latest catalog

CmC
CONNECTICUT microCOMPUTER, Inc.
150 POCONO ROAD
BROOKFIELD, CONNECTICUT 06804
TEL: (203) 775-9659 TWX: 710-456-0052

6502 Bibliography: Part XVII

Dr. William R. Dial
438 Roslyn Avenue
Akron, OH 44320

528. MICRO No. 14, July 1979.

Smola, Paul, "SYM and AIM Memory Expansion." pg. 30.
An easy hardware modification makes MEMORY PLUS a natural for RAMming more data into the SYM and AIM.

Vrtis, Nicholas, "The First Book of KIM—on a SYM", pg. 35-37.
How to modify the programs in this source for the SUM.

Hill, Alan G., "Ampersort," pg. 39-52.
A fast machine language sort utility for the Apple II.

Taylor, William L., "OSI Fast Screen Erase Under BASIC," pg. 53.
This short machine code program fills a need for a fast erase.

Rowe, Mike (Staff), "The Micro Software Catalog: X" pg. 54-56.
Fourteen more 6502 software offerings.

Biles, Noel G., "To Tape or Not to Tape: What is the Question?", pg. 57-59.
Use your scope to examine and diagnose your VIM cassette interface.

Dial, Dr. William R., "6502 Bibliography: Part XI," pg. 61-62.
About 80 new references on the 6502.

529. Personal Computing 3 No. 8, August 1979).

Anon, NCC '79 Report,", pg. 34-36.
Report on the new Apple II Plus, Auto-Start ROM, Apple's Language system (Pascal, etc.), New Apple business software, Apple Graphics Tablet, etc.

530. The Apple Shoppe 1, No. 2 (July 1979)

Anon, "Language Lab," pg. 7-10.
Discussion of the Apple Languages: Basic, Applesoft Basic, Forth Pascal, Pilot, Lisp...Can Fortran and Cobol be far behind? Also how to set up a system to trace one's heritage.

Anon, "Graphics Workshop," pg. 10-12.
Beginning Lo-res and Hi-res graphics.

"Light Pen Applications," pg. 12-13.
Program for taking attendance records.

Anon, "Program of the Month," pg. 13,16.
Program for drawing circuit diagrams.

Anon, "DOS 3.2," pg. 18-19.
Discussion of 3.2 and the new DOS Manual.

531. Southeastern Software Newsletter Iss. 11 (July 1979)

Carpenter, Chuck, "Assembly Language Primer," pg.2-3
Explains how a character is output.

McClelland, George, "SRCH Names File," pg. 4-5
Continuing his interesting series of utilities, the Editor discusses and gives a program for searching the file for names.

Ames, Dave, "Electric Typewriter," pg. 11-12.
A program to work with either the IP-125 or IP-225 printers and will allow you to output text in upper or lower cases.

532. ABACUS Newsletter 1, Iss. 7 (July 1979)

Anon, "Notes on DOS 3.2," pg. 1.
Several tidbits of useful information on DOS 3.2 including how to use the direct command open file.

Anon, "Auto Run Tapes," pg. 1.
How to convert your tapes to Auto-run; very simple!!

Crossman, Craig, "Password,", pg. 2.
How to put a password into your program. Also a siren program to sound on unauthorized attempted entry.

Ford, Bob, "Juggle," pg. 3-4.
Keep as many balls in the air as possible.

Crossman, Craig, "The Hi-Res Corner," pg. 5.
The first of a series of articles on Hi-Res Graphics.

Anon, "UPDATE," pg. 6-7.
The Apple II Business System, the Apple II Plus, Apple's new repair program including diagnostic software and the Modular Parts Exchange Program, description of Apple II PASCAL, etc.

Crossman, Craig, "Program to Disguise your Copyright Notice,"pg. 8.
A short program can be appended to your listing to protect it; and by disguising it, it is harder to wipe out.

Crossman, Craig, "Variable Speed Slow List," pg. 12.
Slow list in any one of 9 selectable Apple speeds.

Anon. 'Bulletin Board Services," pg. 13-14.
A most complete list of Apple Bulletin Boards and CBBS systems.

Freeman, Larry, "Two-Diamonds," pg. 15-15.
A puzzle-type game for the Apple.

533. Creative Computing 5 No. 8.

Friedman, Sol, "A Printer for your PET—For Under $300!"pg. 32-35.
How to use the PR-40 with your PET.

Rhodes, Ned W., "Translating Two-Dimensional Arrays for Integer BASICs," pg. 106-108.
How to add array capability to Apple's Integer Basic.

534. The Paper 2, Iss. 1 (February 1979)

Maier, Gary A., "What Really Makes Your PET Tick?", pg.1-6.
A good tutorial on machine language of the 6502 and PET.

Busdiecker, Roy, "A Decoder Add-On to the MEMEXPLORER," pg. 12-13.
Program allows examination of a block of 20 bytes of PET memory specified by the user.

Buxton, Robert, "Fast-Forward to Find Your Program," pg.14.
DIRECTORY is a program to locate your program on tape.

Wind, Robert H., "Basic in ROM," pg. 16.
Tables listing the addresses where the PET BASIC routines reside.

535. The Paper 2, Iss. 2 (March 1979)

Barroll, Ken C., "Review of the Microtronics M-65," pg. 1.
This unit plugs into two ports in the back of the PET and provides Send and Receive RTTY and Morse.

Busdieker, Roy, "Exploring Pet's Memory: A Real Program," pg. 3-5.
A tutorial on the PET memory and how a program is handled.

Greenup, Campbell Hugh, "How to Address the Screen with These Three Statements—POKE 245, row: PRINT:POKE 266, column," pg. 7.
Explanation of a short PET routine.

Poirer, Rene, "Prevent 'Return Key' Fallout," pg. 10-11.
A fix to prevent dropping out of a program when the return key is accidentally pressed on the PET.

Swan, Warren D., "Change 'Change' (Alien Basic Keyword) to...," pg. 11.
A discussion and explanation of the CHANGE command.

Busdiecker, Roy, "Watch your PET's Wait," pg. 22-23.
An explanation of the WAIT command on the PET.

Busdiecker, Roy, "The Case of the Trigonometric Bug," pg. 12-13.
Tracing down a bug on the PET.

536. The Paper 2, Iss.3 (April 1979)

Simpson, Rick, "An Introduction to Assembly Language Programming," pg. 1, 4-6.
The microprocessor, the PET system, memory organization, ROM and RAM memory, etc.

Landereau, Terry L., "Animation," pg. 18.
A short tutorial on animation.

Julich, Paul M., "Data Files Containing Strings," pg. 19.
All about data files, PET style.

Landereau, Terry L., "Latest Update: Cassette Files," pg. 20-21.
A collection of tricks used to read and write data files reliably.

Landereau, Terry L., "Programming a Flashing Cursor," pg. 21.
Put a cursor in your program.

Busdiecker, Roy, "More About Extended Graphics," pg. 22-23.
How to put graphics on a strip of screen, vertical or horizontal.

Winograd, Fred C., "Application Notes 1 and 2," pg. 24-26.
Two programs for Printers using the CmC ADA 1200 C Adapter.

537. The Paper 2 Iss. 4 (May 1979)

Swan, Warren, "Machine Language Routines for Fast Graphics," pg. 1, 4-10.
Lots of goodies in this tutorial article on PET graphics.

Wachtel, Anselm, "Another Second Cassette Interface," pg. 14-17.
Add a second cassette to your PET.

538. The Paper 2 Iss. 5 (July 1979)

Simpson, Rick, "Introduction to Machine Language," pg. 3-5.
Continuation of this good tutorial.

Busdiecker, Roy, "The Number Game: An Introduction to Computer Arithmetic," pg. 7-8
All about how computers use numbers.

Lee, Arnie, "The Old PET, The New PET and the Blue Sky," pg. 20-25.
All about the new keyboard, the display screen, the cassette drive, the operating system, etc.

539. ABACUS 1, Iss. 1 (January 1979)

Tognazzini, Bruce, "Page by Page List," pg. 3.
List your program page by page.

Anon, "Read and Write to Files," pg. 5.
A program showing how to read and write to disk files.

Danielson, Larry, "Color Killer Mod," pg. 8.
Add this simple mod to your earlier model Apple.

540. ABACUS 1, Iss 2 (February 1979)

Avelar Ed, "Important Addresses and Routines,"pg. 3-6.
Reference chart comparing familiar BASIC commands with the machine language equivalents.

Aldrich, Darrell, "Free Space Program," pg. 11
A short program to show how much free space ramains on your Apple disk.

541. ABACUS, Iss 3 (March 1979)

Avelar, Ed, "Monitor Routines," pg. 5.
Miscellaneous routines for the Apple.

Danielson, Larry, "6 Color Modification," pg. 12.
Convert your early serial number Apple II to six colors, in hi-res graphics.

Shank, Stephen, "Want a Faster Cursor?" pg. 14.
Speed up the cursor or repeat key by a simple hardware mod.

542. ABACUS, Iss 4 (April 1979)

Anon, "Graphics Routines," pg. 2.
Several short programs that can be added to your programs for that extra enchancement.

Wilkerson, David, "Lower-Casing It on the Apple II," pg. 3-4.
A software modification to print in lower case.

Danielson, Larry, "Lower Case Mod," pg. 4-5.
Hardware method of getting your Apple to display Lower Case characters.

Wilkerson, Dave, "Dollars and Cents in Applesoft," pg. 6.
Round off Applesoft to two decimal places.

Yee, Alan, "ASCII Output," pg. 7.
Program outputs ASCII equivalent on request, on the Apple.

543. ABACUS 1, Iss 5 (May 1979).

Anon, "Special Text Output," pg. 3.
Special routines using COUT on the Apple.

Anon, "The WAIT Routine," pg. 5.
All about the WAIT routine for the Apple.

Anon, "Printing Error Messages," pg. 6.
A list of printing error messages.

Anon, "Some Zero Page Explanations," pg. 6.
Tells what each byte in zero page does.

Anon, "Machine Language Program Development Aids," pg. 7.
Many routines in the Monitor can be helpful when developing machine language programs.

Anon, "Apple II Memory Map, Showing Areas Over-Written When Booting DOS 3.1", pg. 8.
Another Memory Map.

Anon, "Color Graphics," pg. 11.
Lo-Res graphics program for the Apple II.

Yee, Dave, "Alphabetizer," pg. 12.
Input names and alphabetize with this program.

Anon, "The Eight Queens Problem," pg. 13.
The Apple searches a solution to put eight queens on a chess board.

544. Dr. Dobbs Journal 4, Iss 7, No. 37 (Aug. 1979)

Colburn, Don, "Those All-Important Extras," pg. 20-26.
A memory display program based on a 6502/CGRS system with EXOS. Also a program written tor a 650X Tim based system with the Per-Sci controller.

Bach, Stephen E., "Disassembler tor Sym-1," pg. 45.
Adaptation ot the 6502 disassembler from Apple for the Sym-1.

545. Stems from the Apple 2, Iss 7. (July 1979)

Hoggatt, Ken, "Ken's Korner," pg. 2
How to put more than one DOS command on one line of the Apple, a handy list ot zero page uses, a novel monitor routine, data and read statements in Applesoft, transparent machine language, etc.

Stein, Dick, "Numerical Sorting in Applesoft," pg. 5-6.
This "QUICKSORT" method is taster than the "BUBBLE SORT."

Porter, Gale, "HEX-HEX-HEX," pg. 7.
Hex numbers are input as strings and output as decimals. Both Integer Basic and Applesoft routines are given.

Newman, Will II, "Text File Build, Store, Retrieve Example," pg. 8.
A tutorial program.

546. The Target (July/August 1979)

Sellars, George, "Statistical Analysis," pg. 2-3.
Several program listings tor the AIM 65 Basic are given.

Riley, Ron, "Basic Hints," pg. 11.
Some advice on using the AIM-65 Basic.

547. Personal Computing 3, No. 9 (Sept. 1979)

Irving, Steve and Arnold, Bill, "Measuring Readability of Text," pg. 34-36.
A PET program to analyze the readability ot Text.

548. Rainbow I, Iss 6. (July 1979)

Simpson, Rick, "Running the Volume 3 Hires Demo on a 32K Apple with DOS," pg. 1.
A simple fix for a problem with the Demo on Vol. 3 ot the Contributed Library, tor the Apple.

Watson, Allen, "Multiply and Divide Subroutines," pg. 2-3.
Discussion of subroutines in the Apple Monitor.

Hirsch, John, "FORTH — or Backwards?", pg. 11-13.
A discussion of this language available tor Pet and Apple.

549. Byte 4, No. 8, (August 1979)

Anon, "Byte News," pg. 89.
Rockwell has introduced a bubble memory board for 128 kbytes of storage which plugs directly into the expansion bus for the AIM-6502 processor (same as for KIM-1), expandable to 16 such memory boards (2 Mbytes).

Appleseed, P.O. Box 68, Milford, NH 03055, pg. 199
Appleseed is a new magazine about to appear, devoted to Apple II sottware.

Information, Unlimited Sottware, 146 N. Broad St., Griffith, IN 46319, pg. 201.
EASYWRITER Is a Word processor tor the Apple II.

Kellerman, David, "Turn your KIM into a Metronome," pg. 213.
Short listing for an adjustable speed metronome.

Allen, Michael, 6025 Kimbark, Chicago, IL 60637, pg. 236.

550. Cider Press 2 No. 4 (August 1979)

Stone, Barney, "Apple Drops RAM Applesott," pg. 5.
Apple has quietly decided to drop the Ram versions ot Applesoft Basic. They will concentrate on Rom Basic which is the version also used in RAM with the new Pascal/Language system. The current version ot the ROM card includes the new Auto-Bool ROM.

Hertzfeld, Andy, "Fix Catalog," pg. 7-9.
The program Fix Catalog, sometimes called Fix Sector Count, corrects the sector count that is printed in the catalog on the Apple disk.

Anon, "Disk of the Month," pg. 2.
The August Disk of the Month includes utilities, games and graphics programs for the Apple II.

Kotowsk, Tom, "Metronome," pg. 9.
A short program for the Apple with speed adjustable with the game paddles.

Frankel, Jeff, "Program Conversion," pg. 9.
A program to change your Integer Basic program to Applesoft and vice-versa. For the Apple II.

Anon, "Memory Chart," pg. 10.
An easy to use memory chart for the Apple.

Silverman, Ken, "Applesoft Interpreler Set," pg. 11.
ROM addresses D000-F7FF giving subroulines entry points, tor the Apple.

Slovick, Linda, "Apple Integer Basic," pg. 12.
Token and Character set tor the Apple Integer Basic.

Anon, "How lo get 21 Hi-Res Color Without Any Hardware Mod," pg. 13.
A software program to glve a lot of hires colors.

Anon, "Variable Delay Atter a Carriage Return," pg. 13.
A program with a bug, submitted by Apple Compuler.

Apple Computer, "Serial Card Handshake Mod," pg. 14.
This is a modification to use the Data Inpul line as a CTS (clear to send) line.

Gannes, Howard; Silverman, Ken; Couch, John, "CHECKBOOK," pg. 15-17.
This program includes the many patches tound necessary and published in many places; for the Apple.

551. KB Microcomputing, No 33 (September 1979)

Feldman, Phil and Rugg, Tom, "Happy Motoring!" pg. 48- 50.
A program to keep track o f tuel consumption, tuel economy, miles driven, etc. For the PET.

DeJong, Dr. Marvin L. "Catching Bugs with Lights," pg. 96-99.
A Hardware approach to debugging with LED monitors.

Downey, Dr. James M. "Make PET Hard Copy Easy," pg. 100-102.
Intertacing ASCII or Baudot Printers to PET's leee bus is a snap with this circuit.

Smith, Darrell G. "Apple II High-Resolution Graphics," pg. 104-106.
All about HiRes on the Apple.

Tulloch, Michael. "Put Your PET on the Bus." pg. 112-115.
With BETSI intertace PET to the S-100 goodies.

Blalock, John M. "Anolher KIM-1 Expansion" pg 130-133.
Packaging the Kim, adding a TTL serial intertace, adding 24K additional memory, etc.

552. MICRO. No 15 (August, 1979)

Bixby, Donald w. "Apple II Serial Output Made Simple" pg. 5-8.
Helpful hints on Implementing Apple II serial output.

Vrtis, Nicholas. "Extending the SYM-1 Monitor," pg. 9-15.
Adds a program relocator, a program listing utility and a trace function.

Morris, E.D.,Jr. "Replace that PIA with a VIA" pg. 17-18.
If your board uses the 6520 PIA, try replacing it with a 6522 VIA to get all the functions of the 6520 plus two timers, a shitt register, Input data latching and a much more powerful interrupt system.

Smith, Ronald C. "PET Cassette I/O" pg. 19.
No more lost files, missing data, etc. with this improved I/O.

Morris, E.D.,Jr. "Tokens" pg. 20.
Discussion of PET Microsolt Basic Tokens.

Bradford, L. William. "A Better LIFE for Your Apple," pg. 22-24.
An enhancement for your LIFE program.

Clements, William C. "EPROM tor the KIM" pg. 25-26.
An easy to build EPROM board requires no special intertacing.

Luebbert, Prof. William F. "What's Where in the Apple," pg. 29-36.
Luebbert's Appte Memory Atlas is very complete, giving the tocation and function of various Peeks, Pokes and Calls and other subroutines.

Rowe, Mike (Staff) "The MICRO Software Catalog: XI," pg. 38.
Reviews tour important programs lor 6502 based micro's.

DeJong, Dr. Marvin L. "Intertacing the Analog Devices 757OJ A/D Converter," pg. 40-41.
Interfacing into together with a demonstration program. For the KIM or other 6502 boards.

Blalock, John M. "SYMple Memory Expansion,"pg. 42-43.
A compact 8K SYM by this hardware Mod.

Zant, Robert F. "Detine HI-RES Characters for the Apple II," pg. 44-45.
A program to easily generate and modify Hi-Res characters on the Apple II.

Zant, Robert F. "Common Variables on the Apple II," pg. 47-49.
Two short routines emulate the Disk II DOS CHAIN capability by allowing the use of common variables under Integer or Applesoft Basic, without a disk.

Dial, William R. "6502 Bibliography: Part XII," pg. 53-55.
Over 115 new references to the 6502 literature are added to the bibliography.

553. PET User Notes 1, Iss 7 (Nov/Dec 1978)

Butterfield, Jim, "Poor Man's D/A Converter," pg. 2
A simple D/A based on a group of resistors.

Church, Rick. "Star Sounds - CB2 Sound," pg. 3.
Sounds for the PET.

Riley, Michael. "Two Player Games with One Keyboard," pg. 4.
Software tor avoiding key lockout.

Bell, John. "GET String Routine," pg. 4.
This routine acts as a substitute tor an INPUT statement.

Buttertield, Jim. "Veritying Tape Loads," pg. 4-5.
Simple verify routine.

Russo, Jim and Chow, Henry. "M7171 Monitor and Merge in High Memory," pg. 6-7.
Routine for the PET.

Russo, Jim and Chow, Henry. "D63777-R63888 (Delete and Resequence)," pg. 7.
A modified routine with line delete capability added.

Cooke, John A. "IEEE Bus Handshake Routine in Machine Language," pg. 8-9.
A routine allowing data transter speeds of over 5000 bytes per second.

Riley, Michael. "Getting Started in Machine Language," pg. 9.
A tutorial for the PET.

Russo, Jim and Chow, Henry. "Merger," pg. 10.
A utility tor the PET.

Seiler, Bill. "PET Renumber 3.0," pg. 12-14.
A useful utility for the PET.

Martinez, Henry. "PET IEEE-488 to SWTPC PR40 Printer Interface," pg. 18.
Hardware for the printer intertace.

Butterfield, Jim. "Memory Usage and Garbage Collection," pg. 18.
Tips on Memory Usage.

Riley, Michael. "Panic Button," pg. 21.
Short machine language routine to help regain control of the cursor.

Butterfield, Jim. "Arrow," pg. 24.
A game for the PET.

554. Call -Apple 2, No. 6. (July/August 1979)

Golding, Val J. "A HEX on Thee," pg. 4-6.
A discussion ot Binary, Hex and different number systems involved in the Apple II. Includes a HEX-DEC Converter Basic program.

Wagner, Roger. "A Fast GR Screen Clear," pg. 8.
Clear the low resotution graphics page of the Apple very fast.

Aldrich, Darrell. "BADR.CREATE," pg. 8.
A program for the Apple to give the start and length of a BLOADed lile.

Aaronson, Tim and Hertzfeld, Andy. "Using Page 2 Text and Lo-Res," pg. 13.
Routines lor special effects on the Apple II.

Anon. "IMA-A new Computing Language," pg. 13.
IMA is a new language by Microversity which allows the use of Integer Basic, Machine Language and Applesoft in the same program.

Garson, David B. "Multiply Demo," pg. 19.
Routine to show the use ot the multiply tunction in the Apple's monitor.

Aldrich, Darrell. "Color Twentyone," pg. 21.
Software approach to creating additional Hi-Res colors.

Golding, Val J. "Hidden Rem Formatter," pg. 21.
Two programs for hidden rams.

Garson, David B. "Soul Searching with the Apple," pg. 22.
A machine language program to go through memory looking for occurences of HEX or ASCII strings that the operator specities. For the Apple.

Kottlnoff, Jett. "Bowling," pg. 24-25.
A well arranged and documented listing tor a game of bowling.

Aldrich, Darrell. "The Apple Doctor," pg. 26.
How to verify a ROM in your computer. Also a discussion of the new AUTO-START ROM and how to put it on the Applesott Firmware Card to achieve optional Autostart action.This way you retain the old ROM and the functions that would have been lost such as STEP, TRACE, etc., that are in the old monitor.

Thyng, Mike. "Applemash," pg. 28.
Discussion of a project to get an IMSAI and the Apple II to talk to each other.

Rivers, Jerry. "Amazing Mystery Program," pg. 30.
A short program tor the Apple.

Creative Computing can help you select the best computer and get the most out of it.

With so many new personal computers being announced and the prices coming down so rapidly, isn't the best bet to wait a year or so to buy a system?

We think not. A pundit once observed that there are three kinds of people in the world: 1) those who make things happen, 2) those who watch things happen and 3) those who wonder what happened. Today, it is those who are getting involved with microcomputers who are making things happen by learning to use computers effectively.

Furthermore, it is not likely that we will see the same dramatic price declines in future years that have already taken place. Rather, one will be able to get more capability for the same price.

The TI-99/4 has excellent color graphics and costs $1150 including color TV monitor.

Which system is for you?

No two people have exactly the same needs. You'll have to determine what capabilities are important to you. Key variables include:

- Upper and lower case. Obviously vital if you are planning to do word processing or anything with text output.
- Graphics. Most systems have graphics but the resolution varies widely. How much do you really need?
- Color. Some systems are B&W, some have 4 colors, others up to 256 colors. Many colors sounds nice, but do you really need 4, or 16, or more?
- Mass storage. The smaller systems are cassette based; larger systems offer floppy disks or even hard disks. What size data bases do you intend to use and is it important to have high-speed random access to an entire data base?
- Languages. Basic is standard but increasingly Pascal, Fortran, Cobol and special purpose languages are being offered.
- Audio, Speech, Music. Are these features important for your planned applications?
- Applications Software. Third party software is widely available for some systems, non-existent for others. Do you need this, or can you write your own?

Unbiased, in-depth evaluations.

At Creative Computing, we obtain new systems as soon as they are announced. We put them through their paces in our Software Center and also in the environment for which they are intended — home, business, or school. We published the first in-depth evaluations of the Texas Instruments 99/4, Atari 800, TRS-80, Ohio Scientific Challenger, Exidy Sorcerer, Apple II disk system and Heath H-8. We intend to continue this type of coverage, not only of systems, but peripherals and software as well.

Sorting: A Key Technique

While evaluations are important, the main focus of Creative Computing magazine is computer applications of all kinds. Many of these require that data be retrieved or sorted. Unfortunately, most programming texts focus on the bubble sort (or straight insertion) and, very infrequently, another technique (usually delayed replacement) and let it go at that.

Yet, except for comparison counting, the bubble sort is the least efficient. Tutorials and articles in Creative Computing demonstrate that the Shell-Metzner and Heapsort are from 50 to 13,000 times as fast as the bubble sort! Consider a sort of 100,000 items on a DEC System 10:

Bubble sort	7.1 days
Delayed replacement	3.8 days
Heapsort	17.3 minutes
Shell-Metzner	15.0 minutes

Needless to say, on a microcomputer, a bubble sort of even 1000 items is agonizingly long.

Free Sorting and Shuffling Reprint

Because sorting and shuffling (mixing a list of items) is so vital in most programming, we are making available a 20-page reprint booklet on Sorting, Shuffling and File Structures along with our May 1979 issue which has several articles on writing user-oriented programs and making the most of available memory space. The reprint booklet and issue are free with 12-issue or longer subscriptions.

At Creative Computing, we believe that computers can be of benefit to virtually every intelligent person in the country. We do not believe that the "Computer priesthood" should confuse and bully the public. As Ted Nelson stated in the Computer Lib Pledge, we do not treat any question as a dumb question, since there is no such thing. We are against computer terms or systems that are oppressive, insulting or unkind, and we are doing the best we can to improve or replace such terminology or systems. We are committed to doing all we can to further human understanding and make computers easy to understand, interactive wherever possible, and fun for the user. The complete Computer Lib Pledge is contained in our May 1979 issue which we are furnishing free to new subscribers.

Free reprint booklet and issue with a new subscription to Creative Computing.

Contributing editor Ted Nelson (L) is author of "Computer Lib/Dream Machines." Publisher David Ahl (R) is a pioneer in computer models, simulations and games.

Computer literacy to everyone

The Creative Computing Software Division is participating with Children's Television Workshop in an important new venture, Sesame Place. These theme parks are being designed to bring interactive computer games and simulations to young children (and their parents) and remove the mystique of computers from the youngest segment of our population. In addition, we are participating in projects with several school systems and museums to write reading comprehension and ecology simulations software. We are also involved in a major college-level computer literacy project.

As a subscriber to Creative Computing, you will benefit from all of these activities. Creative Computing is the Number 1 software and applications magazine. Subscribe today — 12 issues for $15 ($9 saving over the newsstand price). Or, beat inflation and get 36 issues for just $40. Money back if you're not satisfied. Send payment or Visa, Master Charge or American Express number to:

Creative Computing, Attn: Joyce
P.O. Box 789-M
Morristown, NJ 07960

Save time, and call your order toll-free to:
800-631-8112
(in NJ call 201-540-0445)

creative computing

JOIN RAYGAMCO NOW!

Become a member of RAYGAMCO Computer Discount Club.

SAVE 20% AND MORE!

BIG SAVINGS ON EVERY ITEM!

By being a RAYGAMCO Member you receive substantial discounts on every item you purchase, including all hardware, software, accessories, even books and paper! You will also receive a monthly newsletter with all the latest available for your particular computer system, and much, much more — exclusive to RAYGAMCO Members only!

All the famous brand names, including:

APPLE
ATARI
EXIDY/Sorcerer
Kim/Commodore
Alpha Micro
Alpha Pro
Cromemco
Xerox
Soroc
Hazeltine
Sektor
PET
Lear Siegler
Shugart
Texas Instruments

SAVE 20% AND MORE!

Here's how to join.

Fill out the information, and mail. That's all there is to it. Nothing to buy.
I want to be a RAYGAMCO Computer Discount Club Member. Please send my RAYGAMCO Membership card to:

Name ______________________________

Address ______________________________

City ______________ State ______________ Zip ________

Computer (Brand Name) ______________________________

I would like information on (please specify system, part, accessory, book, program, etc.)

____

____

WE HONOR VISA, MASTERCHARGE, BANKAMERICARD.

Store Hours: Sat 10-6, Sun 12-4, Tu-Fri 11-8

TOLL FREE, EXCEPT CA
800-854-6455

RAYGAM, INC.

6791 WESTMINSTER AVENUE WESTMINSTER, CA 92683
TELEX 182274 (714) 891-2587

NOW PRESENTING...

Software for Apple® II

for your Entertainment · Business · Education

Star Attractions:

WRITE-ON Professional Word Processing lets you edit, move, delete, find, change and repeat any body of text, merge and save on disk. Does right-justified margins, centering, page numbering. You can enter name & address into form letters when printing. Edit and merge any text disk file—even files not created by WRITE-ON—and spool text to disk for letter printing or editing. Chain files when printing for an infinite number of pages in a single printer run. Needs Applesoft and 32K.
On Disk with operating manual.......... $99.50

FILEMASTER 2 programs: *FORMAT & RETRIEVAL* comprise a powerful data file manager. Great for everything from phone lists to legal abstracts. Needs 32K. Design your own data structure. Up to 500 characters per record. Up to 15 searchable fields in any combination. On Disk.......................... $34.95

SPACE Multi-faceted simulation of life in interstellar society. You and opponents must make life & death decisions. Keeps track of your progress from one game to next. Needs 48K and Applesoft ROM. Disk.......................... $29.95

Pot D'Gold I or our All New Pot D' Gold II A collection of 49 programs for 16K Apple. Everything from Logic to action games. Only a buck a game. Specify I or II. Price each: Tape $49.... Disk $54

ADVENTURE Fight off pirates and vicious dwarfs. 700 travel options, 140 locations, 64 objects. Needs ROM & 48K. Disk.. $29.95

16K CASSETTE INVENTORY Use item number, description, stock amount, reorder amount, restock date, cost & sell price. Holds up to 140 items. Tape........................ $35

32K DISK INVENTORY: Use stock numbers description, vendor, record of purchase and sales date, amount on hand, cost & sell price, total value. Holds up to 300 items. Disk.............. $40
With Parts Explosion: Disk..................... $50

32K DATA BASE Cross file for phone lists, bibliographies, recipes. Run up to 9 lines of 40 columns each. Search by item anywhere. Disk.................................. $20

24K HI-RES LIFE SIMULATION Conway's equations on 296x180 screen. A mathematical simulation to demo population growth with birth, death and survival as factors. Tape............. $10

16K CIRCUIT LOGIC DEVELOPMENT AID Evaluate circuits of up to 255 gates, including AND, OR, NOR, NAND, XOR, XNOR and INVERTER. Tape.......................... $10

16K MORSE CODE TRAINER Learn Morse Code, and transmit or receive over radio. Tape........................ $10

16K PACIFICA: Discover the floating island and rescue the beautiful princess. To win you must recover the enchanted crown, but you face the threat of magic spells and demons. Tape........ $9.95

RAINBOW'S CASINO 9 gambling games: Roulette, Blackjack, Craps, Horserace, and a few originals that Vegas hasn't heard about. Needs 16K. Tape........................ $29.95

16K SPACE WAR: You in your space capsule battle against the computer's saucer... in hi-res graphics. Tape........... $12

16K MEMORY VERIFY Diagnostic routine to check range of memory. Indicates faulty addresses, data in memory cell, and faulty data. Tape.................................... $5

16K APPLEODION Music synthesis composes original Irish jigs. Enter your own music and save on tape or disk. Includes 3 Bach fugues. Tape.............................. $10

16K APPLEVISION Demo for Hi-Res graphics and music. Tape.................................... $10

32K COMPU-READ 5 programs to teach you speed reading, in stages. Includes synonym and antonym identification. You control your rate of speed, or keep up with the computer's pace. Disk.................................. $24.95

48K PERCEPTION I, II, III random shapes and sizes must be matched. In III, you control format and display time and get weighted scores. Needs ROM. Each Disk............ $24.95

24K POLAR PLOT Plot polar equations in Hi-Res Graphics. Tape.................................... $10

32K SHAPE SCALER Utility to generate and animate Hi-Res graphic shapes. Simple routine provided to inspect position of shapes, and specify precise X/Y coordinates and scale. Needs ROM. Disk.................................. $13.95

APPLE MONITOR PEELED Everything you wanted to know about the Apple Monitor but couldn't figure out. User-written manual in plain English clears your confusion. Only............. $9.95

Don't see what you've been looking for, here? Then write for our FREE SOFTWARE CATALOG. We're saving one just for you!

To order, add $2 shipping. California residents add 6% sales tax. Sorry, we can not ship to P.O. Boxes. VISA/ MASTERCHARGE and BANKAMERICARD Welcomed!

VISA

master charge

RAINBOW COMPUTING INC.

Garden Plaza Shopping Center, Dept. 1A
9719 Reseda Blvd., Northridge, Ca 91324
Telephone: (213) 349-5560

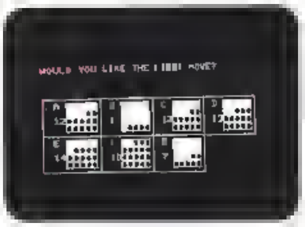
Software for the PET

DOMINOES $ 6.95

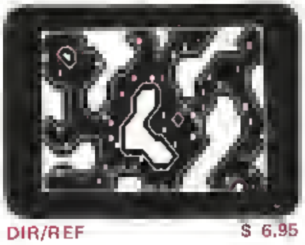
OTHELLO $ 9.95

LETTER SOUARES $ 6.95

RPN MATHPACK $19.95



SUPER NIM $ 6.95



DIR/REF $ 6.95

SPACE WARS $ 9.95

GRAND PRIX $ 6.95

E.S.P. $ 9.95

FORECAST $ 9.95

$ 6.95

BLOCKADE $ 9.95

DEPTH CHARGE $ 9.95

BASKETBALL $ 9.95

SLOT MACHINE $ 6.95

HOME ACCOUNTING $ 9.95

All orders include 3% postage and handling with a minimum of $1.00. California residents include 6% Sales Tax.

VISA MASTERCHARGE

BATTLE SHIP $ 9.95

PET IS A TRADEMARK OF
COMMODORE BUSINESS MACHINES, INC.

SHOOTING GALLERY $ 9.95

PROGRAMMA
INTERNATIONAL, Inc.
3400 Wilshire Blvd.
Los Angeles, CA 90010
(213) 384-0579
384-1116
384-1117

Dealer Inquiries Invited

ZAP $ 6.95

PROGRAMMA Software Program Products